DECEMBER 1968

AD 679269

SYSTEM AND SOFTWARE SIMULATOR

VOLUME I

UNITED STATES ARMY
COMPUTER SYSTEMS SUPPORT
AND EVALUATION COMMAND
WASHINGTON, D.C. 20310

3200.8 (Att 1 to Encl 1)
Mar 7, 66

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) C-E-I-R, Inc. 5272 River Road Washington, D. C.	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

SYSTEM AND SOFTWARE SIMULATOR VOLUME I

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Working Paper

5. AUTHOR(S) (First name, middle initial, last name)

Leo J. Cohen

6. REPORT DATE December 1968	7a. TOTAL NO. OF PAGES 270	7b. NO. OF REFS
8a. CONTRACT OR GRANT NO. DA 49-083 OSA-3306 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

Distribution of this document is unlimited.

11. SUPPLEMENTARY NOTES This documentation has been assembled and released by the sponsoring activity	12. SPONSORING MILITARY ACTIVITY USA Computer Systems Support and Evlauation Command Washington, D. C.

13. ABSTRACT

The System and Software Simulator (S3) is a digital event simulator written in FORTRAN IV and designed to perform simulations of computer systems hardware and software and of the workload being applied to the system.

This and the other three volumes constitute the complete documentation available on S3. Volume I describes the inputs, outputs, methods and capabilities of S3. Volume II contains the flowcharts, narrative description of the flowcharts, layouts and descriptions of the tables utilized by S3. Volume III contains descriptions of the assembly language used for preparation of input to S3, of the macro capability of the assembler, and of the modifications made to S3 to provide additional output data. Volume IV is the program documentation on the internal workings of the assembler. It consists of table descriptions, flow charts and narratives, and file descriptions.

These volumes are a collection of documentation delivered under two separate contracts. They have not been edited and as such are considered working papers.

DD FORM 1 NOV 65 1473

Security Classification

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Simulation Computer Systems Operating Systems Evaluation Timing Dynamic Event Simulation						

Security Classification

WORKING PAPER

The documentation on the System and Software Simulator (S3) contained in this and the other three volumes is considered a working paper and no claims are made as to its accuracy. There has been no attempt to edit the information. Discrepancies and inconsistencies are known to exist.

This information is being released as a service to interested parties and to satisfy the numerous requests for information on S3.

The documentation of S3 is contained in four volumes. Volumes I and II are contract end items delivered under contract number DA-49-083 OSA-3306 and contain the technical descriptions of S3. Volume I describes the inputs, outputs, methods and capabilities of S3. Volume II contains the flowcharts, narrative description of the flowcharts, layouts and descriptions of the tables utilized by S3.

Volumes III and IV contain the documentation delivered as contract end items under contract number DAAB09-68-C-0118. Volume III contains descriptions of the assembly language used for preparation of input to S3, of the macro capability of the assembler, and of the modifications made to S3 to provide additional output data. Volume IV is the program documentation on the internal workings of the assembler. It consists of table descriptions, flow charts and narratives, and file descriptions.

WORKING PAPER

WORKING PAPER

TM CEIR-S3

SYSTEM AND SOFTWARE SIMULATOR

S3

September 15, 1967

SUMMARY

This technical manual explains the methods and capabilities of the System and Software Simulator, S3. The S3 is a program designed to accept process and specialize certain data to represent a particular computing system. The resultant simulation model program then accepts executive instructions whose "object time" is the simulated time of the system's activity. The first data set is called configuration data and is analogous to microprogramming instructions. The simulation model thus derived operates like an actual computer by interpreting and executing instructions.

Prepared By:

C-E-I-R, INC.
5272 RiverRoad
Washington, D. C.

WORKING PAPER

TABLE OF CONTENTS

SECTION	DESCRIPTION	PAGE
7.15.2	Value	7-10
7.15.3	Location	7-10
7.16	MOVE	7-11
7.16.1	Card Type	7-11
7.16.2	Number Characters	7-11
7.17	MOVE-E	7-11
7.17.1	Card Type	7-11
7.17.2	Number Characters	7-11
7.18	COMPUTE	7-11
7.18.1	Card Type	7-11
7.18.2	Number Instructions	7-12
7.18.3	[illegible] Number	7-12
7.19	MATH	7-12
7.19.1	Card Type	7-12
7.19.2	Number Adds	7-12
7.19.3	Number Multiplies	7-12
7.19.4	Number Divides	7-12
7.20	OPEN	7-13
7.20.1	Card Type	7-13
7.20.2	File Number	7-13
7.20.3	Input/Output	7-13
7.21	CLOSE	7-13
7.21.1	Card Type	7-13
7.21.2	File Number	7-13
7.21.3	[illegible]	7-13

APPENDICES

[illegible]

ILLUSTRATIONS

SECTION I

INTRODUCTION

1.0 GENERAL

A digital event simulation of a computing system's performance measures it's success in terms of the accuracy with which the simulation approximates a real system's performance. To achieve acceptable accuracy it is necessary for the digital event simulation to provide representation for four specific items:

A. Hardware

B. Users

C. Operating System

D. Files

In providing the model required for a properly accurate simulation, it is necessary that a high detail of representation be made available. In particular, it is required that all of the performance characteristics of each of the hard wired equipments, as well as of the system's software, be represented. It is necessary that the dynamic structural components, specifying topology, have a presence in the simulation as well. The performance of this simulation model is dependent solely upon the accuracy of the data presented to it.

1.1 HARDWARE

Hardware description for purposes of simulation encompasses a description of peripheral devices; transmissions, memories and the CPU's. The hardware configuration description is independent of any dynamic topological connections, and is strictly functional, relating to the performance properties of the equipments involved. A brief description of the performance properties to be represented for the peripheral devices is explained below.

1.2 PERIPHERAL DEVICES

There are two types of peripheral devices of interest in a computing system. These are sequential, and random access.

A. The sequential type includes:

1. Card readers

2. Paper tape readers

3. Line printers

4. Magnetic tapes

B. Random access includes:

1. The random access disc and drum

2. The card random access devices

1.2.1 Sequential Devices

The sequential device specifications of interest for the purposes of simulation are time to transmit a record, the time required to traverse an inter-record gap, and any penalty time that may be associated with time dependent, sequential Input-Output (I/O) relative to the particular device. Thus, on a magnetic tape, the transfer of a specified block of data occurs at a given rate. The time to traverse enough tape to pass to the next block is also a necessary specification, and finally STOP START time and any associated penalty time for exceeding the STOP START interval must be represented.

1.2.2 Random Access

For random access devices there are two basic timing considerations to be considered. Certain devices require a seek time, which is independent of the data transfer time involved. As in the case of a sequential device however, once data is transferred it is transferred in a block unit, and again there may be INTER-RECORD time that might be considered to be equivalent to the latency time on a rotating random access device.

1.3 TRANSMISSIONS

The transmissions that are specified for simulation represent the connection between the peripheral devices and memories of the system. There are two types of transmissions used for simulation; the wire and the channel. A transmission has a method of data transfer, such as bit-parallel or byte-sequential.

1.3.1 Wire

For a wire to transmit such a unit there will be required an amount of Central Processing Unit (CPU) activity.

1.3.2 Channel

A channel is capable of transmitting a specified collection of such units without any CPU assistance. Thus the channel exhibits simultaneous operation with the system's CPU, and the wire does not.

Specification of transmission performance for simulation purposes therefore requires a statement of the data transfer unit, the rate at which transfer is accomplished, and the type of the transmission. In addition, it is necessary to specify the point at which the total I/O transmission rate saturates the available memory cycle time for a given memory module.

The performance properties of interest for a module of memory is its size, in addressable units. For simulation, interest centers on the separately powered memory module so that it may be isolated to take account of the effect of memory interference on available CPU access time. The specification of the simulation model requires that the number of such separately powered memory modules be given.

The performance characteristic of the CPU's of the system is the rate at which it is able to execute various classes of instructions. These instruction classes are for add, multiply and divide type instructions. As in the case of the specification of memory modules for a simulation, it is necessary to specify as well, the number of CPU's.

Therefore, S3 requires a performance specification for each of the elements to be simulated within the above described four categories. Additionally, information is required in order to specify the interconnections between peripheral devices and memories, and between memories and CPU's. This topological interconnection of information, and the performance specification, forms the configuration data for the simulation model. Thus the simulator machine provides the potential for expression of virtually any type of system configuration for the purpose of simulation.

EXAMPLE:

Configuration data specifying two CPU's and two memories may be further specified to interconnect CPU's and memories such that each CPU has access to both memories. This would represent a multiprocessor configuration, such as the GE 600 series. Also two CPU's and two memories may specify a 1401-7090 Main-Satellite system, by specifying that each of the CPU's is connected to one and only one of the memories.

1.4 USER

The various hardware equipments for which the configuration data described above provides performance and topological specification is loaded by the users of the system in terms of the amount of time required

by programs on these several hardware resources. These resources are loaded, as well, by the operating system. Since the operating system's function is to distribute resources to the user's programs, this will be explained separately. In order to distinguish between user and operating system programs, we shall refer to the former as worker programs.

A worker program, prior to its execution, represents a potential load on the computer system's resources. The program itself requires memory space, and its execution, CPU time. Futhermore, its I/O activity will refer to files resident on peripheral devices available in the system and these devices are reached by the specified transmissions. The use of a system's facilities by worker programs is referred to as potential because the actual use of facilities is dynamically determined at the execution time of the worker program itself. Therefore, in simulation it is necessary to represent the several potential load possibilities that a worker program may exert on the simulated system. This is accomplised by allowing each of the worker programs whose performance in the system is to be simulated, to be represented by what we shall refer to as a worker program graph.

1.5 WORKER PROGRAM GRAPH

The program graph is defined as a collection of directed edges and nodes. Edges that leave a node are referred to as exit edges, and those that enter a node as entering edges. An exit edge is associated with two values. The first of these two values is the probability of choosing this exit edge on leaving the node. Thus, if there are two exit edges from a given node and the probability of taking one edge is .30, then the probability of taking the second edge would be .70. (figure 1-1). The second value associated with an edge is a "statement".

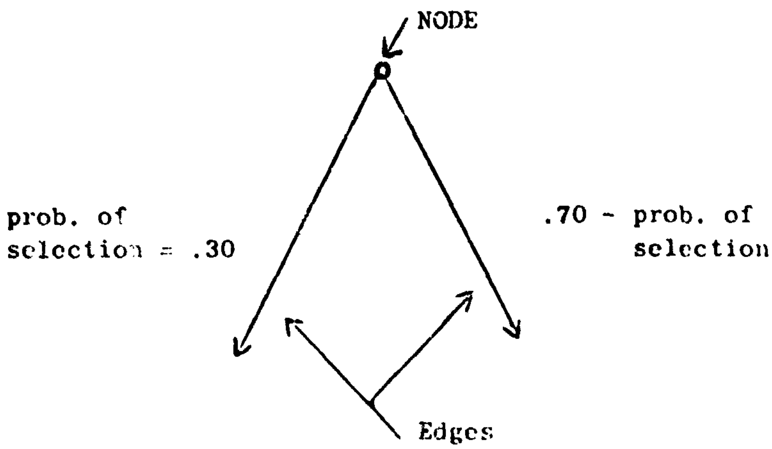

Figure 1-1. EDGES AND NODES

The collection of nodes and edges of a program gra[illegible]e to represent the possible directions that the logic of a worker program [illegible] follow during its execution. Nodes represent junction points for these possibilities, the edges represent the operations that the worker program would carry out if the edge were selected at execution time. Thus, the program graph, as a collection of nodes and edges, is comparable to the process chart, which is usually the first stage of an actual programming effort.

1.6 EDGE STATEMENTS

Edge statements to indicate the possibilities that the simulator makes available for representation of the potential load exerted by a worker program on the simulated system follows.

The statements may be divided into three categories such as:

A. CPU utilization

B. I/O utilization

C. Control

A list of edge statements designed for specification of CPU utilization are in Table 1-1. Each of these statements represent a way of expressing a requirement for the CPU, in terms of the logical operation exercised by execution of the edge with which it is associated.

Table 1-1. EDGE STATEMENTS, CPU UTILIZATION

MOVE	N	move N core memory data units on the mainframe
MOVE-E	N	move and edit N core memory data units
COMPUTE	N	choose a mix of instructions relative to the number N
MATH	A, B, C	execute A adds, B multiplies and C divides

1.6.1 MOVE Statement

The MOVE statement causes the simulator machine to determine the number of instructions that would have to be executed in order to accomplish the movement of the N data units. This number of instructions is then translated into an amount of CPU time required to execute according to the specification given in the configuration data.

1.6.2 MATH Statement

The MATH Statement allows the worker program to specify the number of add, multiply and divide operations involved in the computation of a formula, or collection of formulas. The simulator machine interprets each of these parameters into a number of instructions to be executed and then interprets the result into the time required forthe CPU to execute, using the configuration data specification for the CPU. This is accomplished in conjunction with a Gibson Mix formula to indicate execution memory access overlap.

1.6.3 COMPUTE Instruction

The COMPUTE instruction provides the creator of a worker program with capability for expressing a CPU requirement in lieu of any more specific knowledge of the activity expected for the program.

1.7 I/O UTILIZATION STATEMENTS

The utilization statements made available by the simulator are explained in the following text. Because there is an application of S3 for which it is highly desirable that the worker programs be device independent, the techniques employed by the simulator reduce the number of I/O statements to four. These four statements are listed in Table 1-2. The associated parameter is an ordinal file number. This number is local to the program graph within which it appears and identifys that file within the program.

Table 1-2. I/O STATEMENTS

OPEN	A	open ordinal file A
CLOSE	A	close ordinal file A
READ	A	read next record from ordinal file A
WRITE	A	write next record to ordinal file A

When an I/O statement is encountered by the simulator machine in the course of a simulated execution of a worker program graph, the ordinal file number is used to locate the simulated file on the peripheral device on which it resides. The simulated operating system determines the order of access by this worker program to the device, and when the simulated I/O operation is initiated, the simulator machine computes the time required to find the file on the device, and to transfer a block of information from the device over the selected transmission to the memory.

1.7.1 OPEN Statement

The S3 provides a mechanism for the user to specify, among other things, the number of buffers to be retained in the memory for each of the ordinal files referenced by a worker program. The purpose of the OPEN statement is to initiate the filling of all buffers for the ordinal file designated.

1.7.2 CLOSE Statement

The CLOSE statement closes the designated file by emptying the buffers via simulated output operations.

1.7.3 CONTROL Statements

Table 1-3 gives the several control statements that the simulator makes available and by means of which the creator of a worker program graph may specify conditional and unconditional transfer loops through portions of the graph the call of subroutines, and finally termination of the simulated execution of his worker program.

Table 1-3. CONTROL STATEMENTS

TRANS	A	transfer to statement A
TRANS-P	A,B	with probability A, transfer to statement B
LOOP	A,B	transfer to statement A, as a loop, B times, after which, go in
EOF	A	transfer to statement A on end-of-file

continued

Table 1-3. (Continued)

SUBR	A	execute the subroutine starting at statement A
EXIT		return from subroutine
TERM		terminate execution of the worker

Of the several control statements listed in Table 1-3 the End of File (EOF) statement may be unfamiliar. The specification of a file (to be covered later) includes its size. When enough reads for an input file, or writes to an output file, have been encountered during the simulated execution of the worker program to equal the number of data units in the file, the simulator machine indicates this by setting a flag for that file. If the user elects to use the end-of-file condition in the program, it is done with the EOF statement. If this statement is not used then the condition of the end-of-file flag, as established by the simulator machine, has no effect on the simulated execution of the worker program.

With the edge statements that S3 makes available to its users for the specification of worker program graphs, it should be clear that the process chart of an actual worker program has a close identification with this representational form for simulation. An example of this is shown in the process diagram (figure 1-2). The program flow for diagram (figure 1-2) is shown in figure 1-3, and figure 1-4 shows the "programming" of the program flow using the simulator language.

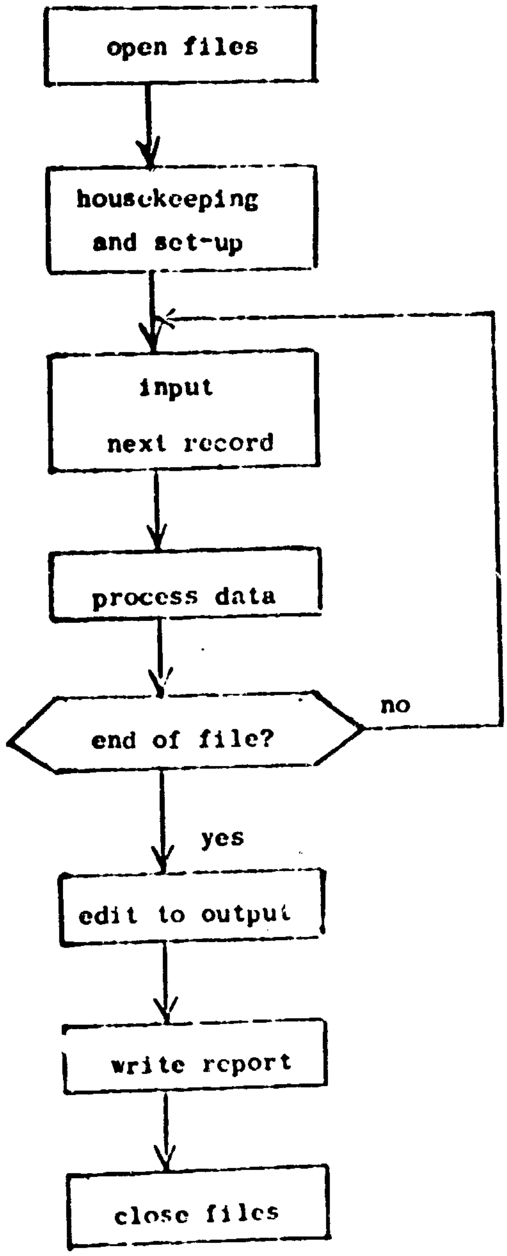

Figure 1-2. PROCESS DIAGRAM

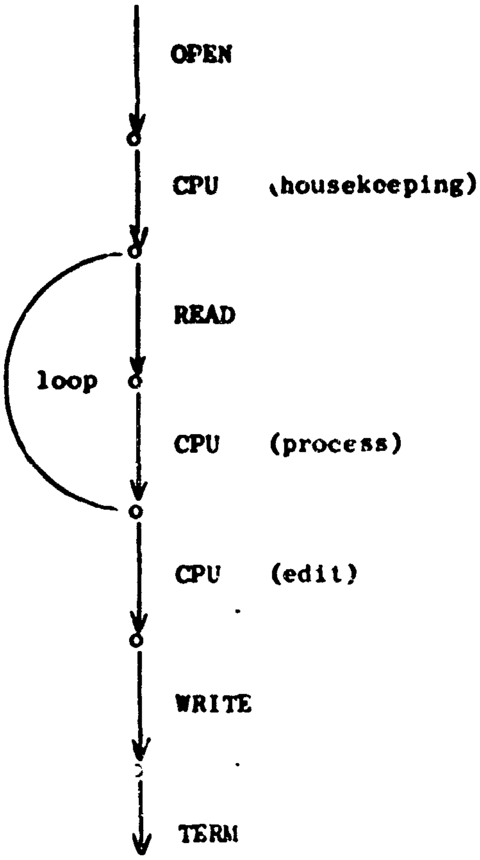

Figure 1-3. PROGRAM FLOW

```
          OPEN      1

          OPEN      2

          COMPUTE   1500

NEXT      READ      1

          MATH      600, 30, 10

          COMPUTE   4000

          EOF       END

          LOOP      NEXT, 100

          MOVE-E    3000

          WRITE     2

          CLOSE     1

          CLOSE     2

          TERM
```

Figure 1-4. PROGRAMMING THE PROGRAM FLOW

1.8 THE OPERATING SYSTEM

One common characteristic for an operating system and worker programs is that they are both programs. Each can be represented by flow charts, and for simulation purposes by process charts. This common characteristic of the techniques used for the representation of a worker program via the program graph and simulator machine statements, are suitable for the representation of an operating system. The problem of drawing a process chart, and the program graph for an operating system brings up one of the basic differences between an operating system and worker program. Specifically, the logic of the execution of the worker program depends on the data on which it operates. The logic of the execution of an operating system depends on the needs of the particular worker programs to which it is distributing the system's facilities. Consequently, the items on which the operating system must perform its functions are the worker programs themselves, or, more accurately, representative names for these worker programs. Therefore, if worker programs A, B, and C are resident in the simulated memory of a one CPU system, and all three of them are available for execution seeking the CPU facility, it is the operating system's responsiblity to determine which of the three will next gain access to this facility, and to defer the execution of the remaining two. By this method A may be selected as the current operating program and be given the CPU facility, to continue its simulated execution. Simultaneously, the operating system must defer the execution of programs B and C, by placing their names in a list or queue indicating deferrment.

1.8.1 Additional CPU Statements

It is therefore necessary to augment the collection of CPU edge statements to provide the necessary capabilities for the complete specification of an operating system program graph. Table 1-4 lists additional S3 statements and explains each application. Careful review of these statements indicates that any operating system can be successfully represented for simulation, ranging from the simple batch processor, to multiprogramming operating systems for a single CPU, multiprocessing operating systems for multiple CPU's and operating systems for the control of multi-computers.

1.8.2 Interrupt Technique

To complete an operating system specification a technique must be available for the representation of interrupts. S3 provides a general method for the simulation of interrupts through the system. This is accomplished by means of an interrupt vector in the simulator machine which is a list of simulator machine statements that have been specified by the user in the preparation of the operating system graph.

Table 1-4. ADDITIONAL S3 STATEMENTS

MEMORY	A,B,C	used to query the current state of the memory map. The parameters specify conditions for the query.
ALLOCATE	A,B	used to assign memory and modify the memory map
DEALLOCATE	A,B	reverses the functions of ALLOCATE
PACK	A,B	to modify the memory map without allocation
EXAMINE	A,B	examine queue A according to criteria B
PLACE	A,B	place program or I/O call A into queue B
SELECT	A,B	select program or I/O call A from queue B
BUFFER		used to control buffering
SEEK		used to locate a named file on its device
IO READY	A	determine the executability of the next I/O call in queue A
IO ADVANCE		initiate a given I/O operation
IO TERM	A	search queue A for an I/O termination condition
INTERRUPT	A,B	interrupt CPU number A for Condition B
DISABLE	A	disable interrupts A
ENABLE	A	enable interrupts B
CLOCK	A	set clock to time A

The user may associate each item of the interrupt vector with any statement of his choice, but the statement selected will generally be TRANS with its parameter specifying a particular location in the operating system graph. The general method for I/O interrupt handling is as follows: after completion of a simulated I/O operation, the simulated execution of the current operating program is interrupted and the next simulator machine statement to be executed is selected from the interrupt vector item associated with the I/O interrupt that has occurred. Therefore, if the statement in the selected interrupt vector item is a transfer into

a particular portion of the operating system graph (most likely this will be to the interrupt processing portion) then the simulation of operating system action on the occurrence of a particular I/O interrupt is initiated.

NOTE

> The operation of the interrupt function of the simulator machine resembles that of all real systems in that the current execution state of the interrupted current operating program is retained for that program, so that when next selected for resumption of its execution, that execution is continued from the point of interrupt.

1.9 GENERAL CONCEPT

The techniques thus far described invest the simulator machine with properties that echo the logical characteristics of an actual computer. Thus, configuration data associates the simulator machine with a specific set of peripheral devices, transmissions, memories and CPU's. The worker and operating system program diagrams, in terms of simulator machine statements, then appear to the simulator machine, and are acted upon by it, in precisely the same way that an actual computer accepts and executes actual programs. The important point is that this is the general characteristic of the simulator machine. By means of configuration data and program diagrams particular characteristics are specified for the system in whose simulation we are interested. Thus, this general characteristic that S3 has for identifying, virtually completely, with actual systems is specialized to identify a particular model with its actual counterpart. The result is an accuracy that can only be matched by the observed performance of the actual system itself.

1.10 FILES

Files belong to the individual worker programs but reside on the hardware peripheral devices of the system. As mentioned earlier, the worker programs specify file references in terms of the ordinal naming system already mentioned. The files that are so referenced are given representation in the simulation model as part of the configuration data.

1.10.1 File Structure

The structure of the file on the peripheral device is defined in terms of blocks. The block is the unit of transfer between the device and memory, by the assigned transmission. A block is defined in terms of a data unit specification, the data type (alpha or numeric), whether or not the data is packed, and any code conversion requirements that might be necessary.

1.10.2 Memory Block

In the memory, the block consists of a number of records. The specification in the configuration data for the number of records is called the blocking factor, and each READ or WRITE I/O statement refers to a record within a block that is associated with the file by its ordinal name. This means that the record is the unit of data processed by the simulated worker program so that each successive read simulates the delivery of the next record to the worker program for processing. Records are delivered on call to the worker program until all records in the simulated block have been removed. The BUFFER statement may be used in the simulated operating system to ascertain when this occurs and to select the next action that the operating system undertakes in order to refill the block from the file.

By specifying that the blocking factor is one record per block and that the number of buffers for the execution of a particular worker program is one, a buffered execution of that program can be simulated. By raising the blocking factor and/or increasing the number of blocks assigned to a particular ordinal file, varying degrees of buffering may be simulated for a particular execution.

1.10.3 Buffering Control

The control of any buffering that is allowed is properly an operating system function. The simulator machine statements made available for representation of the operating system are clearly sufficient to allow for inclusion of any and all buffering control techniques that might be of interest.

1.11 CONCLUSION

The method devised for the simulator machine, as broadly described in this section, achieves several major objectives that are fundamental to an accurate and exhaustive simulation of an actual computer configuration - plus - operating system. These objectives are:

A. The complete performance and inter-connection specifications for the hardware of the system are accomplished in terms of the configuration data.

B. The potential load on the system, in terms of the system facilities required by all worker programs that may be run on it and the operating system required to run them, is represented by means of coded program graphs using simulator machine statements.

C. The files on which the various simulated programs operate are represented in the particular simulation model by means of a structure specification, a device residence specification, and a specification of the conditions of buffering allowed.

To make the application of S3 as general as possible, the technique has been developed to provide machine independence in the simulated worker program. This is accomplished for INPUT-OUTPUT by means of the ordinal naming approach for files. It is accomplished for CPU utilization by means of simulator machine statements whose operands specify the amount of work to be done in terms of data units, allowing the simulator machine to translate this into the number of instructions required to accomplish the specified function according to the particular capabilities of the mainframe under simulation. This means that a given collection of worker programs, once prepared for simulation, may be applied to an arbitrary range of simulated computer systems quite independently of the peripheral device configuration, the number of memories, CPU's and transmissions that an individual system might include.

1.11.1 Major Differences

The major difference from system to system in so far as the worker programs are concerned is the operating system itself. Thus, each computer system that is simulated is associated with its own operating system. A review of the worker program statements made available by the simulator machine will show that they are not only independent of the simulated computer configuration within which they are executed, but are as well independent of the operating system, relying upon the logical structure of the operating system and the specification of the interrupt vector to accomplish the necessary interface with the simulated configuration.

Finally, because we have separated the specification of file structure, location, and buffering from the specifications for both the general hardware configuration and the worker programs themselves, these general parameters of the file environment of a particular execution on a given system may be varied without any modification to the other simulation elements being required.

SECTION II

SIMULATION MODEL DESCRIPTION

2.0 WORKER ROUTINE MODULE

The worker routine module is comprised of five worker routine base statements (type 81-85) and 18 worker routine statements. The logic of the simulation model is such that once worker routines have been written, they do not require modification when used in conjunction with different machine configurations.

<u>EXAMPLE</u>: The same set of worker routines could be used as input when simulating an IBM 1401, or a GE 625.

2.1.1 Worker Routine Base Statements

The worker routine base statements contain information that will be used during the pre-simulation phase of the simulation model. The following list includes information contained in the worker routine base statements.

A. Identification for simulation and statistical purposes

B. Ordinal file and real file identification

C. Memory requirements

D. Incidence of generation

E. Priority

F. Generation Limit

G. Generation Intervals

2.1.2 Worker Routine Statement Interpretation

The worker routine statements will be interpreted during the running of the simulation to represent the program represented by a given worker routine. The logic of the simulation model considers

worker routines to be re-entrant, and further differentiates between a passive routine and active program. This means that a worker routine whose first incidence of arrival is seventy minutes in a simulation that runs only one hour, will never be considered a program. Although it has been residing in the simulation model for the length of the simulation, it has not been introduced into the active flow of programs that are being run through this simulation model.

2.1.3 Worker Routine Statement Logic

The logic of the worker routine statements provides an easily understood and accurate tool for representing existing programs as input to the simulator. Further, the range of these statements is sufficiently broad to allow the representation of proposed programs which, though not written, have definable limits associated with them.

2.2 OPERATING SYSTEM MODULE

Twenty-eight operating system statements, plus the interpretative worker routine statements are available to transcribe the logic of an operating system into input for the simulation models.

2.2.1 Modular Design

The modular design of the simulation model is such that an operating system may be easily modified, or completely changed between runs without the necessity of major alterations to the other modules of the simulation model.

2.2.2 Operating System Design

The design of the operating system module will allow the user to literally transcribe their logic charts of an operating system into statements acceptable as input to the simulation model.

2.3 CONFIGURATION DATA MODULE

The configuration data is the means by which the hardware characteristics of a given computer may be entered as input data to the simulator. The configuration data is used to describe the physical characteristics of the following:

A. Central Processors

B. Memories

C. Channels

D. Devices

E. Control Units

The physical relationship between the listed items may be accurately described and the scope of information that may be included allows the most complex computer to be accurately described. Elementary computer configurations may be easily represented by omitting fields within a statement or entire statements that are not required to fully describe the configuration. The modularity associated with the simulation model allows the reconfiguration of computers without the need to drastically modify the other segments of the simulation model.

2.4 SYSTEM PARAMETER MODULES

The system parameters are the liaison between the worker routines, operating system(s) and configuration data. It is entirely possible that worker routines, operating systems and configuration data may be collected as input by three separate groups. It is essential that the personnel compiling the three sets of inputs be responsible for the system parameters. A listing of the descriptive information contained in the system parameters is as follows:

A. A description of the files, including their length, seek time (if pertinent), position on a device, and device identification.

B. Dump Control information for debugging a simulation model.

C. Statistical output control information for defining the type of statistics to be gathered.

D. Simulation model control information - to limit the length of simulation and the number of statistical outputs to be gathered in that time period.

2.5 PRE-SIMULATION MODULE

This portion of the model is responsible for receiving the input data and building an internal structure from this data for the simulation executive allowing highly efficient execution once the

actual simulation is started. The configuration data and system parameters are accepted first. This input will be used to set table dimensions and these tables are located in memory. The tables will contain information on queues, memory maps, channel availability, etc. Once the tables have been constructed, the operating system and worker routine statements are accepted as input and stored in their respective tables. It is at this point that absolute addresses are supplied for all transfer statements. When all input data has been received and table starting addresses have been located for the simulate executive, the pre-simulation module terminates itself and turns control over to the executive portion.

2.6 SIMULATION EXECUTIVE MODULE

The executive is responsible for the actual running of the simulation once all input has been accepted and formatted into the proper tables. The operations of the executive have been kept to a minimum in an attempt to make a simulation run as efficient (in time used) as possible. In essence, the three major functions of the executive are to:

A. Interpret subroutine calls

B. Check for diagnostics

C. Determine when statistical output is to be produced

2.6.1 Routines and Subroutines

The routines and subroutines that are available to the executive are listed below.

A. Timing routine

B. Continuous Interrupt routine

C. Program Generation routine

D. File/Device Maintenance routine

E. Channel/Control Unit Availability routine

F. Dump/Trace/Snap Diagnostic routines

G. Statistical Analysis routine

H. All Worker and Operating System statement subroutines

I. A Vector Table to locate all segments of the simulation model

SECTION III

LIMITS AND RESTRICTIONS

3.0 GENERAL

This section describes the limits and restrictions of the Simulation Model for various entities that may be represented in a given simulation. If any of these values are exceeded, a diagnostic is generated and the simulation terminated.

3.1 CPU

The limit on the number of CPU's that may be represented in a simulation is 5.

3.2 MEMORIES

The limit on the number of memories that may be represented in a simulation is 30. No more than 10 separate memories may be associated with one CPU.

3.3 PAGES

The limit on the number of pages that memories may be divided into is 1000.

3.4 CONTROL UNITS

The limit on the number of control units that may be represented in a simulation is 50.

3.5 CHANNELS

The limit on the number of channels that may be represented in the simulation is 50. When the tally of channels represented in a given simulation is made by the simulation model, a value of one is associated with a selector channel and a value of two is associated with a multiplexor channel. Therefore one selector channel and one multiplexor channel will equal three units. The simulation model always compares the number of units represented by the different types of channels

against the limit of 50. No more than 20 control units may be associated with one CPU.

3.6 DEVICES

The limit on the different types of devices that may be represented in a simulation is 50. The limit on the total number of devices of all kinds represented in a simulation is 100.

3.7 FILES REAL/ORDINAL

The limit on a number of ordinal files that may be represented in a simulation is 800. The limit on the number of real files that may be represented in a simulation is 150. The limit on a number of real files associated with one device in a simulation is 15. The final statistics will include statistics on 21 INPUT-OUTPUT files per worker routine.

3.8 QUEUES

The limit on the number of queues represented in a simulation is 30. These may contain a total of 2000 queue entries.

3.9 FUNCTIONS

The limit on the number of functions that may be represented in a simulation is 100.

3.10 AVAILABILITY TABLE

The limit on the number of entries in the availability table is 200. Each channel/control unit path to a devide is considered one entry.

3.11 LOAD CLASS ENTRY

The limit on the number of load class entry statements that may be included in the configuration data for a simulation is 15.

3.12 RUN CLASS ENTRY

The limit on the number of run class entry statements that may be included in the configuration data for a simulation is 5.

3.13 SIMULATION CONTROL

If, when multiplied together, the two fields of the simulation control statement (S/P card code 31) produce a value greater than 24 hours (1440 minutes) the simulation is terminated.

3.14 INTERRUPT

The limit on the number of interrupts that may be associated with each CPU being represented in a simulation is 20. The first 8 of these 20 interrupts are pre-assigned and must reference addresses within the operating system.

3.15 W/R STATEMENTS

The limit on the number of worker routine statements that may be represented in a simulation is 15,000. Any number of operating systems, primary worker routines or worker routines may be represented within this limit.

3.15.1 W/R Programs

The limit of the number of programs that may be active within the simulation at a given time is 300.

2.15.2 W/R Inputs/Outputs

The limit on the number of INPUT-OUTPUT activities that may be in progress coincidentally is 900.

3.16 SWITCHES

The limit on the number of switches that may be referenced in a simulation is 200. Switches numbered 101-200 are considered to be global and may be used as liaison between operating systems. Each of the 5 CPU's permitted in a simulation may have 20 local switches (1-20) associated with it.

3.17 MARK

The limit on the number of mark counters that may be accumulated during a simulation is 10. These mark counters are referenced by the second parameter of the compute statement. Only operating systems or primary worker routines may make use of this facility.

SECTION IV

DEVICE DEFINITION AND CONFIGURATION DESCRIPTIONS

4.0 GENERAL

This section is a detailed device definition and configuration description for the system. It includes keypunch instructions and sample configuration cards.

NOTE

All fields which are not used must be zero filled. No alpha characters may be punched in these cards except in columns 76-80.

4.1 CPU DEFINITION CARD Ø1

4.1.1 Card Code Ø1

4.1.2 CPU Identification Number

This field may contain any four digit number from ØØØ1 to 9999.

4.1.3 CPU Logical Data Unit

The primary unit of data processed by this CPU. For example, in the IBM System 360 the Logical Data Unit is the BYTE.

4.1.4 Decimal Arithmetic Times

These fields must be filled if the CPU being defined is capable of performing base 10 arithmetic.

A. Number of Decimal Digits

When determining the time to perform a decimal arithmetic operation the number of digits operated upon must be supplied.

B. Decimal Add Time

The time, expressed in microseconds, to add two fields of the length given above.

C. Decimal Multiply Time

The time, expressed in microseconds, to multiply two fields of the length given above.

D. Decimal Divide Time

The time, expressed in microseconds, to divide one number of the length given above into a number twice the length above giving a quotient of length above, and a remainder of the same length.

4.1.5 Fixed Point Arithmetic Time

These fields must be filled if the CPU being defined is capable of performing base 2 arithmetic.

A. Fixed Point Add Time

The time, expressed in microseconds, to add two fields of the standard length, taking both fields from main storage and returning the result to storage, including all load and store operations.

B. Fixed Point Multiply Time

The time, expressed in microseconds, to multiply two fields of the standard length, taking both fields from main storage and returning the product to storage.

C. Fixed Point Divide Time

The time, expressed in microseconds, to divide one standard length field by another, taking both fields from storage and returning the quotient or the remainder to storage.

4.1.6 Floating Point Arithmetic Times

These fields must be filled if the CPU being defined is capable of performing floating point arithmetic.

A. Floating Point Add Time

The time, expressed in microseconds, to add two single precision floating point numbers, including pre and post-normalization if required or standard, taking both fields from storage and returning the result to storage.

B. Floating Point Multiply Time

The time, expressed in microseconds, to multiply two single precision floating point numbers, taking both fields from storage and returning the product to storage.

C. Floating Point Divide Time

The time, expressed in microseconds, to divide one single precision floating point number into another, taking both fields from storage and returning the quotient to storage.

4.1.7 System ID

This field may be used for card identification or any user function. It is not checked or used by this system.

NOTE

The limit on the number of CPU's that may be represented in a simulation is 5.

4.2 CPU DEFINITION CARD Ø2

4.2.1 Card Code Ø2

4.2.2 CPU Identification Number

This field may contain any four digit number from ØØØ1 to 9999.

4.2.3 Instruction Length

The average instruction length expressed in Logical Data Units.

4.2.4 Decimal Digits Per Logical Data Unit

The number of decimal digits which can be placed in a Logical Data Unit; e.g., two decimal digits can be placed in ne BYTE in the IBM-360.

4.2.5 Characters Per Logical Data Unit

The number of characters which can be placed in a Logical Data Unit; e.g., six alphaneumeric characters can be placed in a 36-bit word in the UNIVAC11Ø8.

4.2.6 Move Time Per Logical Data Unit

The time, expressed in microseconds, to move one Logical Data Unit.

4.2.7 Move and Edit Time Per Character

The time, expressed in microseconds, to move and edit a single character.

4.2.8 Fixed Point Table

The Fixed Point Table should be completed if the CPU being described is capable of base 2 arithmetic.

A. Entries

Each entry in the fixed point conversion table has two values associated with it. The first value is a number of Logical Data Units; the second is a number of decimal digits. The purpose of this table is to describe the size of the fixed point arithmetic which can be manipulated by this CPU. For example, the IBM 360 is capable of doing half word, full word, and double word fixed point arithmetic. To describe half word arithmetic, assuming a BYTE is the Logical Data Unit, the entry would be | Ø2 | Ø4 |. This means that using two Logical Data Units (BYTES), the System 360 can represent any 4-digit decimal number expressed in base 2. This value is arrived at by taking:

Two BYTES X 8 bits per BYTE = 16 bits

16 bits - 1 sign bit = 15 bits

$2^{15} - 1 = 32767_{(10)}$ which, rounding down, allows expression of 4 decimal digits.

The next entry in this table would be | Ø4 | Ø9 | which describes full word fixed point operations in System 360.

4.2.9 Floating Point Table

The Floating Point Table should be completed if the CPU being described is capable of floating point arithmetic.

A. Entries

Each entry in the floating point conversion table has two values associated with it. The first value is a number of Logical Data Units; the second is a number of decimal digits. The purpose of this table is to describe the size of the floating point fields which can be manipulated by this CPU. For example, System 360 is capable of doing short and long precision floating point arithmetic. To describe short precision floating point, assuming a BYTE is the Logical Data Unit, the table entry would be | Ø4 | Ø7 | . This means that using 4 Logical Data Units (BYTES), System 360 can represent a floating point number to seven decimal digits of precision. This value is arrived at by taking a 32-bit word (4 BYTES) less a 1 bit for characteristic sign, less a 7-bit characteristic, leaving a 24-bit fraction.

$2^{24} - 1 = 16{,}777{,}215$ which, rounding down permits 7 positions of precision.

The next entry in this table would be | Ø8 | 17 | which describes double precision floating point arithmetic.

4.2.10 Gibson Mix Factor

This field contains a value that represents an average mix of instructions for a commerical application. This value will be applied to the number of instructions associated with the COMPUTE statement to produce a simulated time value.

4.2.11 System ID

This field may be used for card identification or any user function. It is not checked or used by this system.

4.3 MEMORY DEFINITION CARD Ø3

4.3.1 Card Code Ø3

4.3.2 Memory ID

This field may contain any 4 digit number from ØØØ1 to 9999.

4.3.3 Memory Access Unit

This entry describes the number of usable bits (not including parity or other check bits) which are accessed by a single memory reference. For example, the IBM 1401 memory access unit is 6 bits.

NOTE

Neither the word mark nor the parity bit is counted.

4.3.4 Memory Cycle Time

Memory Cycle Time is the minimum time, expressed in microseconds, between the start of a memory access and the start of the next memory access to the same memory unit.

4.3.5 Memory Access Time

Memory Access Time is the minimum time, expressed in microseconds, between the start of a memory access and the availability of the data accessed.

4.3.6 Memory Size in Memory Units

Memory Size is the number of memory access units in this memory.

NOTE

The Memory Access Unit is not necessarily the same size as the Logical Data Unit. For example, in the IBM System 360 Model 30 the Memory Access Unit is the BYTE, and the Logical Data Unit is also the BYTE. However, in the Model 40, the Memory Access Unit is the half word (2 BYTES) while the Logical Data Unit is still the BYTE.

4.3.7 Page Size in Memory Units

Page Size is the Unit in which memory is allocated, usually the size of the Memory-Protect Bounds.

NOTE

The limit on the number of memories that may be represented in a simulation is 30. Not more than 10 memories may be associated with one CPU.

The limit on the number of pages that may be represented in a given simulation is 1000.

4.4 CHANNEL DEFINITION CARD Ø4

4.4.1 Card Code Ø4

4.4.2 Channel ID

This field may contain any 4 digit number from ØØØ1 to 9999.

4.4.3 Channel Type

This field is used to describe the channel as either a selector or a multiplexor channel. A selector channel is capable of operating one and only one devide at a time. A multiplexor channel is capable of operating more than one device at a time.

4.4.4 Type Codes

1 = Selector Channel

2 = Multiplexor Channel

4.4.5 Maximum Selector or Burst Mode Transfer Rate

If the channel being described is a selector channel, enter the maximum transfer rate in characters per second.

If the channel being described is a multiplexor channel which is capable of a greater transfer rate when operating a single device (Burst Mode) than when operating multiple devices, enter the maximum transfer rate when operating a single device.

4.4.6 Selector or Burst Mode Interference Rate

If this channel shares CPU time with the main processor, enter the percent of CPU time lost per 1000 CPS transfer rate for this channel. If the Interference Rate is not a linear function, choose the average load point to determine this value.

4.4.7 Maximum Multiplexor Transfer Rate

This entry should be supplied only if the channel being described is a Multiplexor Channel. Enter the Maximum Transfer Rate (CPS) when this channel is operating in multiplexor mode.

4.4.8 Multiplexor Interference Rate

Enter the Interference Rate for this channel when it is operating in multiplexor mode as shown above.

4.4.9 System ID

This field may be used for card identification or any other user function. It is not checked or used by this system.

NOTE

The limit on the number of channels that may be represented in a simulation is 50. No more than 20 channels may be associated with one CPU. A multiplexor channel counts for two when a tally of channels is made against the limit.

4.5 DEVICE DEFINITION CARD Ø5

4.5.1 Card Code Ø5

4.5.2 Device Identification Number

This field may contain any 4 digit number from ØØØ1 to 9999.

4.5.3 Device Type

This field is used to describe the device as one of the following:

A. Unit Record

B. Tape

C. Random Access

D. Other

4.5.4 Data Transfer Rate (All Devices)

The Data Transfer Rate is the rate at which the device is capable of receiving or transmitting data.

4.5.5 Transfer Width (All Devices)

The Transfer Width is the number of bits which are transferred as a unit to this device, excluding parity or other check bits. For example, a 7-track tape has a width of 6 bits, a 9-track tape has a width of 8 bits. Note however, if the device or control unit translates the data, such as a 7-track tape drive on System 360 using the translate

feature, into a greater or lesser number of bits, the device should be described in terms of the number of bits accepted, not the number of bits recorded, i.e., a 7-track tape using the translate feature would be defined with a width of 8 bits. A 7-track tape using data conversion would be defined with a width of 6 bits.

4.5.6 START/STOP Time (Any Device)

START/STOP Time is the total time, expressed in microseconds, over and above the transfer time which the device requires to be activated and deactivated. For example, a System 360 2400 9-track model 1 tape drive has a START/STOP time of 16,000 microseconds.

NOTE

> During START/STOP time, the channel, control unit, and device are all busy.

4.5.7 Device Time

Unit Record Device Time is the time, expressed in microseconds, during which a BUFFERED unit record device is busy after the channel is free.

NOTE

> During unit record device time, the channel is considered free after the data transfer is completed, even though the device and control unit are still busy.

4.5.8 Tape Rewind Time (Tape Devices Only)

Tape Rewind Time is the amount of time, expressed in hundredths of a minute required to rewind a tape.

NOTE

> During rewind time the device is considered to be busy, but the control unit and channel are free.

4.5.9 Time Limit Before Invoking a Penalty (Any device)

Certain devices are capable of providing faster I/O times than normal if the time between I/O operations does not exceed a certain limit. For the purpose of simulation, we consider the fastest possible time, the normal time. Then, if the time limit entered in this field is exceeded between I/O calls to this device, the time penalty is added to the normal I/O time.

4.5.10 Penalty Time

The time, expressed in microseconds, which is to be added to the normal I/O time if no I/O call to this device has occurred during the time limit described above.

Penalty Time is added to the channel, control unit, and device time.

4.5.11 Form Advance Time

This field may be used when a printer is being described. The START/STOP time in the sixth field is normally used to represent a single line advance. If it is desired to represent multiple line form advance, the time in microseconds must be placed in this field.

NOTE

> The limit on the number of devices that may be represented in a simulation is 100.

4.6 CPU CONFIGURATION CARD Ø6

4.6.1 Card Code Ø6

4.6.2 CPU Number

When configuring a system, there will be one CPU configuration card per CPU in the system. The CPU cards must be numbered sequentially in this field from ØØØ1 to 9999.

4.6.3 CPU Identification Number

This field must contain the CPU Identification Number on the CPU definition card for this CPU.

4.7 MEMORY CONFIGURATION CARD Ø7

4.7.1 Card Code Ø7

4.7.2 Memory Number

When configuring a system there will be one memory configuration card per memory in the system. The memory cards must be numbered sequentially in this field from ØØØ1 to 9999.

4.7.3 Memory Identification Number

This field must contain the memory identification number from the memory definition card which defines this memory.

4.7.4 CPU's Associated with this Memory

The CPU number (not Identification Number) of each CPU which can reference this memory.

4.7.5 Channel Configuration Card Ø8

4.7.6 Card Code Ø8

4.7.7 Channel Number

When configuring a system there will be one channel configuration card per channel in the system. The channel cards must be numbered sequentially in this field from ØØØ1 to 9999.

4.7.8 Channel Identification Number

This field must contain the channel identification number from the channel definition card which defines this channel.

4.7.9 CPU's Associated with this Channel

The CPU number (not Identification Number) of each CPU which can reference this channel.

4.8 CONTROL UNIT CONFIGURATION CARD Ø9

4.8.1 Card Code Ø9

4.8.2 Control Unit Number

When configuring a system there will be one control unit configuration card per control unit in the system. The control unit cards must be numbered sequentially in this field from ØØØ1 - 9999.

4.8.3 I/O Code

The I/O code specified the possible directions of Data flow through this control unit.

1 = Input only
2 = Output only
3 = Input and Output

4.8.4 Channels Associated with this Control Unit

The channel number (not Identification Number) of each channel which can address this control unit.

NOTE

> A control unit must be assumed for each channel, even though one does not exist in a real configuration. Type 9 and type 10 statements are used in conjunction to build on availability table. The maximum number of entries permitted in this table is 200. Each control unit/channel path to a device is considered one entry.

4.9 DEVICE CONFIGURATION CARD 1Ø

4.9.1 Card Code 1Ø

4.9.2 Device Number

When configuring a system there will be one device configuration card per device in the system. The device cards must be numbered in this field from ØØØ1 to 9999.

4.9.3 Device Identification Number

This field must contain the device identification number from the device definition card which defines this device.

4.9.4 Seizing Code

This code is used to specify whether this device may be seized by a worker routine or not.

1 = Seizable
2 = Non-Seizable

4.9.5 Control Units Associated with this Device

The control unit number (not Identification number) of each control unit which can address this device.

NOTE

A control unit must be assumed, and named, even though one does not exist in a real configuration.

4.10 TO-FROM TABLE CARD 11

4.10.1 Card Code 11

4.10.2 To-From Table:

A To-From Table should be filled out whenever there is more than one file resident on a device. Each row and column in the table is identified by a relative file location as shown below:

		FROM		
		File 1	File 2	File 3
	File 1	13	47	12
TO	File 2	07	64	19
	File 3	87	41	93

The average time to go from one file to another on the same device is entered into the table. For example, to go from relative file 2 to relative file 3 on the device specified above would take 41 milliseconds in addition to the normal I/O time.

4.10.3 Device Number

This file identifies the device with which this To-From Table is to be associated.

4.10.4 To Row

This entry specifies which row of the To-From Table this card defines.

4.10.5 Array Dimension

This field specifies the number of rows and columns in this table.

4.10.6 Seek Times in Milliseconds

These fields contain the To-From Table entries in milliseconds.

4.11 FUNCTION TABLE CARD 12

4.11.1 Card Code 12

4.11.2 Functions

Functions are generalized input/output expressions which are used to simulate rare input/output operations. A function is defined by the amounts of time, expressed in microseconds, the channel control unit, and device are busy.

4.11.3 Function Number

The number by which this function is identified in a function statement.

4.11.4 Channel Time

The time the channel is busy executing this operation.

4.11.5 Control Unit Time

The time the Control Unit is busy executing this operation.

4.11.6 Device Time

The time the Device is busy executing this operation.

4.11.7 Device Number

The number (not Identification Number) of the device on which this I/O function is to be performed.

NOTE

The limit on the number of functions that may be represented in a simulation is 100.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LEO COHEN ASSOCIATES

Application: DEFINITION CARD LAYOUTS by ________ Date 4/24/67 Job No. CSD Sheet No. ____

CPU (card C1)

Field	Columns
Card code (C1)	1–2
C.P.U. ID	3–6
CPU logical data unit (BITS)	7–10
DECIMAL ARITHMETIC TIMES IF AVAILABLE: # of dec. digits	11–12
Add time (μ sec)	13–19
Multiply time (μ sec)	20–26
Divide time (μ sec)	27–33
FIXED POINT ARITHMETIC TIMES: Add time (μ sec)	34–40
Multiply time (μ sec)	41–47
Divide time (μ sec)	48–54
FLOATING POINT ARITHMETIC TIMES: Add time (μ sec)	55–61
Multiply time (μ sec)	62–68
Divide time (μ sec)	69–76
SYSTEM ID.	77–80

CPU (card C2)

Field	Columns
Card code (C2)	1–2
C.P.U. ID	3–6
[illegible]	7–10
[illegible]	11–12
Move time per logical unit	13–19
Move and edit time per char	20–26
FIXED POINT TABLE: Entry 1, Entry 2, Entry 3, Entry 4	27–42
FLOATING POINT TABLE: Entry 1, Entry 2, Entry 3	43–54
GIBSON MIX FACTOR	55–62
SYSTEM ID.	77–80

MEMORY (card C3)

Field	Columns
Card code (C3)	1–2
Memory ID	3–6
Memory [illegible] unit (BITS)	7–10
Memory cycle time in microseconds	11–17
Memory access time in microseconds	18–24
Memory size in memory units	25–33
Page size in memory units	34–39
SYSTEM ID.	77–80

CHANNEL (card C4)

Field	Columns
Card code (C4)	1–2
Channel ID	3–6
Maximum [illegible] (CPS)	7–14
[illegible] K words	15–20
Maximum multiplexor [illegible] (CA)	21–27
[illegible] K words	28–33
SYSTEM ID.	77–80

DEVICE (card C5)

Field	Columns
Card code (C5)	1–2
Device ID	3–6
Type	7
Transfer rate in units per second	8–14
Transfer unit (BITS)	15–19
Start + stop time (μ sec)	20–26
Unit record device time	27–33
Tape rewind time (MIN.)	34–36
Time limit before [illegible] penalty	37–43
Penalty time	44–50
FORM ADVANCE TIME	51–57
SYSTEM ID.	77–80

NOT REPRODUCIBLE

C/D

Form X24-6599-0
Printed in U. S. A.

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Company: LEO COHEN ASSOC.
Application: CONFIGURATION DATA
by ______ Date ______ Job No. CSD Sheet No. ______

Card	Columns 1–2	Columns 3–6	Columns 7–10	Columns 11–80
CPU	CC 06	CPU #	CPU ID	
MEMORY	CC 07	MEM #	MEM ID	CPU'S ASSOCIATED WITH THIS MEMORY (fields 12–15, 16–19, 20–23, 24–27, 28–31)
[illegible]	CC [illegible]	CHNL #	CHNL ID	CPU'S ASSOCIATED WITH THIS CHANNEL (fields 12–15, 16–19, 20–23, 24–27, 28–31)
[illegible]	CC [illegible]	CTL #		I/O; CHANNELS ASSOCIATED WITH THIS CONTROL UNIT (fields of 4 columns, 12–75)
DEVICE	CC 10	DEV #	DEV ID	CONTROL UNITS ASSOCIATED WITH THIS DEVICE (fields of 4 columns, 12–75)
(blank)				

SECTION V

SYSTEM PARAMETERS AND DESCRIPTIONS

5.0 TO FROM TABLE STATEMENT

This system parameter statement is used to construct TO-FROM tables for each device represented in a simulation. The values inserted into these tables represent the seek time required to access a particular file on a device relative to the last file accessed on that device.

5.1.1 Card Code 21

5.1.2 Device Number

This field may contain any four digit number from 0001 to 9999.

5.1.3 TO ROW

This two digit field must contain the number of the TO ROW to which the following seek times pertain.

5.1.4 Array Dimension

This two digit field must contain the size of the table required to contain the number of files being represented on a given device.

5.1.5 Relative File Number

These fifteen fields of five digits are used to represent the seek time from one file to another on a given device.

NOTE

A. A TO-FROM STATEMENT must exist for every device being represented even though only one file is associated with that device.

B. The number of TO-FROM TABLE TIME entries must equal the value in the ARRAY DIMENSION field.

C. No more than 15 files may be associated with a device.

D. The seek time associated with the relative files must be in milliseconds.

5.2 FUNCTION STATEMENT

5.2.1 Card Code 22

5.2.2 Function Number

This field may contain any four digit number from 0001 to 9999. It is associated with the corresponding number in the first parameter of a FUNCTION statement as used by an operating system.

5.2.3 Channel Time

This seven digit field represents channel usage time in microseconds. There is an assumed decimal point after the fourth digit.

5.2.4 Control Unit Time

This seven digit field represents control unit usage time in microseconds. There is an assumed decimal point after the fourth digit.

5.2.5 Device Time

This seven digit field represents device usage time in microseconds. There is an assumed decimal point after the fourth digit.

5.2.6 Device Number

This four digit field represents the device (by number) to be used by the function statement named in the second field.

5.3. QUEUE STATEMENT

5.3.1 Card Code 23

5.3.2 Queue Number

This field may contain any four digit number from 0001 to 9999.

5.3.3 Maxirum Number Entries

This four digit field represents the maximum number of entries permitted in a given queue.

5.3.4 Queueing Method

This two digit field represents the queueing method to be used by the simulation model when inserting and removing entries.

1 = FIFO (first in, first out)

2 = LIFO (last in, first out)

3 = Priority (a parameter of the GENERATE statement or the SYSTEM DEFINITION statement)

5.3.5 Queue Control

This two digit field represents the type of transaction item being stored in a given queue.

1 = AT (available transaction)

2 = IOT (I/O transaction)

NOTE

If the queue control parameter of this statement does not match the first parameter of the PLACE and SELECT statements, an error message will be generated and the simulation will be terminated.

The limit on the number of queues in a simulation is 30. There may be a maximum of 2000 queue entries at any given time during a simulation.

5.4 LOAD CLASS STATEMENT

This system parameter is used to designate which of the central processors being represented are able to load a given worker routine. The information contained in the LOAD CLASS statement will be used in conjunction with information from the RUN CLASS statement (Card Code 25) and the PROGRAM DISTRIBUTION statement (Card Code 28) to determine where routines may be loaded and, once loaded, under which central processor they will be run.

5.4.1 Card Code 24

5.4.2 Load Class Entry Number

This field specified the number of this LOAD CLASS statement. There may be a total of fifteen LOAD CLASS statements associated with a given simulation.

5.4.3 CPU 1 Through 5

These five fields of two digits each specify, for a given LOAD CLASS, which central processors may load a worker routine. The five fields allow the maximum number of central processors permitted in a simulation to be represented.

Example 1:

Consider a configuration with three CPU's associated with one memory and agroup of worker routines that can be run under the control of any of the central processors.

One LOAD CLASS statement will be required and will appear as follows:

CARD CODE	=	24
LCE #	=	1
CPU #	=	1
CPU #	=	2
CPU #	=	3

Example 2:

Consider a configuration with CPU 1 associated with Memory 1 which representsa 7094 and CPU 2 associated with Memory 2 which represents a 1401.

Two LOAD CLASS cards will be required and will appear as follows:

CARD CODE	=	24	CARD CODE	=	24
LCE #	=	1	LCE #	=	2
CPU #	=	1	CPU #	=	2

NOTE

> This system paramaters must contain at least one LOAD CLASS statement.
> The limit on the number of load class entries in a simulation is 15.

5.5 RUN CLASS STATEMENT

This system parameter indicates to the simulator which central processor can run a program that has been loaded into a given memory.

5.5.1 Card Code 25

5.5.2 RUN CLASS Entry Number

This four digit field identified each of the RUN CLASS entries that may exist in a given simulation.

5.5.3 CPU 1 through 5

These five fields of two digits will associate central processors with the RUN CLASS ENTRY #.

Referring to EXAMPLE-2 in the LOAD CLASS statement, the following two RUN CLASS statements will be required.

CARD CODE	=	25	CARD CODE	=	25
RCE #	=	1	RCE #	=	2
CPU #	=	1	CPU #	=	2

These statements will, in effect, describe the configuration being simulated. They will indicate, to the Simulator Model, the following:

a. A Worker routine loaded by CPU #1 can be run by CPU #1 or CPU #2 (if logic permits).

b. A Worker routine loaded by CPU #2 can be run by CPU #1 or CPU #2 (if logic permits).

c. A Worker routine loaded by CPU #3 must be run by CPU #3.

NOTE

The system parameters must contain at least one RUN CLASS statement.

The limit on the number of run class entries in a simulation is 5.

5.6 FILE DESCRIPTION STATEMENT

5.6.1 Card Code 26

5.6.2 Real File Number

This field contains the real file number and may be any four digit number between 0001 and 9999.

5.6.3 Device Number

This four digit field specified the device with which the real file is associated.

5.6.4 Relative Location

This two digit field specified the relative location on a device of the file being described. For example, if real file 8 and 9 were the only two files on a disc, their relative positions on that device would be 1 and 2, if there is only one file per device, a 1 must be inserted into this field.

5.6.5 Buffer Length

This seven digit field specifies a file buffer length in characters.

5.6.6 Records Per Buffer

This four digit field specifies the number of records contained in one of a files buffers.

5.6.7 Number of Buffers in File

This seven digit field specifies the number of buffers (or blocks) in a file, it is in essence a description of the file length.

NOTE

There must be a file description statement for each file referenced during a simulation. The limit on the number of real files that may be represented in a simulation is 150.

5.7 SYSTEM DEFINITION STATEMENT

5.7.1 Card Code 27

5.7.2 Standard Priority

This four digit field specifies the priority number to be associated with all worker routines that have not been assigned a priority by their GENERATE statement. Since Ø1 is reserved for at least one operating system, this number must be in the range Ø2-99.

5.7.3 Memory Allocation Scheme

This three digit field represents the manner in which the Simulation Model will allocate memory.

1 = contiguous instruction pages, non-contiguous data and I/O pages.

2 = non-contiguous instruction pages, contiguous data and I/O pages.

3 = contiguous instruction pages, contiguous but separate data and I/O pages.

4 = contiguous instruction pages, data and I/O pages.

5 = non-contiguous instruction pages, non-contiguous data and I/O pages.

5.7.4 CPU-1 O/S Program Number

This four digit field specifies to the simulation model the number of the operating system to be associated with CPU-1.

The following four fields in this statement are used for the same purpose, depending upon the number of CPU's being represented in a given simulation.

NOTE

The limit of CPU's that can be represented in a given simulation is five.

5.8 PROGRAM DISTRIBUTION STATEMENT

This system parameter indicates (in a multi-CPU configuration) which central processor may receive a worker routine and which central processor(s) may run that worker routine.

5.8.1 Card Code 28

5.8.2 Program Number

This four digit field identifies the worker routine to be associated with the information in the rest of this statement.

5.8.3 Receiving CPU

This three digit field indicates the Central processor that will receive the worker routine associated with this statement.

5.8.4 Load Class Entry Number

This three digit field specifies which LOAD CLASS statement is to be referenced to determine which central processor(s) may load the worker routine named in the first field.

NOTE

There must be a PROGRAM DISTRIBUTION statement for each worker routine in a given simulation.

5.9 DUMP CONTROL STATEMENT

5.9.1 Card Code 29

5.9.2 Dump Table 1 Through 25

The first twenty-five fields (two digits each) of this statement allow the user to specify which tables are to be dumped at pre-determined

intervals and at a simulation termination. The following value in any of these fields controls which tables are to be dumped.

0 = dump
-1 = no dump

5.9.3 Trace

Activating the trace field of this statement will cause a trace line to be written everytime an instruction is executed. It is activated in the same manner as the dump table fields.

5.9.4 Snap

The snap field is used in the same way that the trace field is used. In addition to the information supplied by a trace, will be added the identification of the active COT, AT and IOT (if one exists).

NOTE

The interval at which dumps are taken is controlled by the SIMULATION CONTROL statement, which is described below.

If a diagnostic occurs during the pre-simulation handling of input data, the dump control statement is disregarded and all dump tables are written to tape.

5.10 STATISTICS CONTROL STATEMENT

5.10.1 Card Code 30

5.10.2 Statistics Table -1 through 10

These ten fields of two digits each will specify which STATISTICS TABLES are to be developed at the time intervals specified by the SIMULATION CONTROL statement.

The following value in any one of those fields controls which STATISTICS TABLES are to be developed.

0 = develop statistics
-1 = do not develop statistics

5.11 SIMULATION CONTROL STATEMENT

5.11.1 Card Code 31

5.11.2 Statistics Interval

This four digit field specifies in minutes the simulated time interval between statistical outputs and/or dumps.

5.11.3 Total Number of Intervals

This four digit field specifies the total number minus one, of statistical outputs and/or dump to be taken during a given simulation. A statistical output and/or dump is automatically taken at simulation termination.

NOTE

If the values in the two fields of the statement when multiplied together are greater than 1440 minutes (24 hours) an error message will be generated and the simulation model will terminate.

5.12 INTERRUPT VECTOR STATEMENTS

There will be an interrupt vector associated with each CPU being represented in a given simulation. This interrupt vector has 20 entries. The first 8 entries are pre-assigned and will be used for the following:

1 = I/O termination
2 = Read/Write
3 = Function
4 = Clock
5 = Receive
6 = Program termination
7 = End of seek
8 = Open/Close

The first eight entries in the interrupt vector of each CPU being represented must contain a transfer address to the appropriate location within the operating system. The remaining 12 entries in the interrupt table vector are available to be used as the configuration of the operating system dictates. The interrupt vector table is represented by two interrupt vector statements, each of which contain ten fields.

5.13 INTERRUPT VECTOR STATEMENT -1

5.13.1 Card Code 32

5.13.2 CPU Number

This four digit field specified the CPU to be associated with this interrupt vector table.

5.13.3 O/S Program Number

This four digit field specifies the operating system to be associated with this interrupt vector table.

5.13.4 Interrupt Address Number 1 through Number 10

The ten (four digit) fields that follow must contain eight interrupt addresses and may contain two optional interrupt addresses.

5.14 INTERRUPT VECTOR STATEMENT-2

5.14.1 Card Code 33

The format of this statement is identical to interrupt vector statement 1 excepting the card code. If an operating system requires 10 or less interrupts this card may be omitted.

NOTE

> The limit on the number of interrupts that may be associated with one CPU is 20. The first eight interrupts must have addresses and the twelve remaining interrupts are available to the user at his option.

5.15 O/S MEMORY ALLOCATION STATEMENT

5.15.1 Card Code 34

5.15.2 O/S Program Number - Memory Number

This system parameter consists of five paired fields of four digits each. It is used to specify in which memory, a particular operating system is to be loaded. There must be one paired entry for each operating system being simulated. The first field of the pair will designate the operating system program number and the second paired field will specify the memory number with which that operating system is to be associated. The simulator will always load an operating system into the low order pages of memory.

5.16 POISSON FUNCTION STATEMENTS

The following two system parameters will allow the user to replace the Simulator's Poisson function values with values of their own choice.

5.16.1 Number 1 Poisson Definition Statement

5.16.2 Card Code 35

The rest of this statement consists of ten (five digit) fields to specify a value for the argument (o through 45) of the Poisson function.

5.16.3 Number 2 Poisson Definition Statement

5.16.4 Card Code 36

The ten (five digit) fields of this statement will specify values for the second half of the Poisson function (50 through 95).

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LEO J COHEN ASSOC.
Application: SYSTEM PARAMETERS
by ______ Date ______ Job No. ______ Sheet No. 1

TO-FROM TABLE CARD (TIME IN MILLISECONDS) — SEARCH NOT SEEK IF ENTRY NEGATIVE

Columns	Field
1–2	Card code: 21
3–6	Device #
7–8	To unit
9–10	Array dim
11–15	Relative file # 1
16–20	# 2
21–25	# 3
26–30	# 4
31–35	# 5
36–40	# 6
41–45	# 7
46–50	# 8
51–55	# 9
56–60	# 10
61–65	# 11
66–70	# 12
71–75	# 13
77–80	SYST. I.D.

FUNCTION CARD

Columns	Field
1–2	Card code: 22
3–6	Function #
7–13	Chained time µ sec
14–20	Control unit time µ sec
21–27	Device time µ sec
28–31	Device #
77–80	SYST. I.D.

QUEUE CARD

Columns	Field
1–2	Card code: 23
3–6	Queue #
7–10	Max. no. items
11–12	QUEUE METHOD
13–14	QUEUE CONTROL
77–80	SYST. I.D.

LOAD CLASS CARD

Columns	Field
1–2	Card code: 24
3–6	Load class entry #
7–16	CPU #
77–80	SYST. I.D.

RUN CLASS CARD

Columns	Field
1–2	Card code: 25
3–6	Run class entry #
7–16	CPU #
77–80	SYST. I.D.

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LEO J. COHEN ASSOC.
Application: SYSTEM PARAMETERS by ______ Date ______ Job No. ______ Sheet No. 2

26 — FILE DESCRIPTION CARD

Columns	Field
1–2	CC (26)
3–6	REAL FILE #
7–10	DEVICE #
11–12	REL LOC
13–19	BUFFER LENGTH (CHARS)
20–23	RECS @ BUFF.
24–30	BUFFERS @ FILE
31–75	FILE DESCRIPTION CARD
76–80	SYST. I.D.

27 — SYSTEM DEFINITION CARD

Columns	Field
1–2	CC
3–5	STD. PROG.
6–8	ALLOC
9–12	CPU-1 O/S PROG #
13–16	CPU-2 O/S PROG #
17–20	CPU-3 O/S PROG #
21–24	CPU-4 O/S PROG #
25–28	CPU-5 O/S PROG #
29–75	SYSTEM DEFINITION CARD
76–80	SYST. I.D.

28 — PROGRAM DISTRIBUTION CARD

Columns	Field
1–2	CC (28)
3–5	PROG #
6–8	← RECEIVING CPU
9–11	← LOAD CLASS ENTRY
12–75	PROGRAM DISTRIBUTION CARD
76–80	SYST. I.D.

29 — DUMP CONTROL CARD

Columns	Field
1–2	CC (29)
3–52	DUMP TABLE # (1–25): 1 … 25
53–54	S…
55–56	Trace
57–75	DUMP CONTROL CARD
76–80	SYST. I.D.

30 — STATISTICS CONTROL CARD

Columns	Field
1–2	CC (30)
3–22	STATISTICS TABLE (1–10): 1 … 10
23–75	STATISTICS CONTROL CARD
76–80	SYST. I.D.

31 — SIMULATION CONTROL CARD

Columns	Field
1–2	CC (31)
3–6	STAT. INTER. (MIN)
7–10	TOTAL # INTER.
11–75	SIMULATION CONTROL CARD
76–80	SYST. I.D.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LEO J COHEN ASSOC.

Application: SYSTEM PARAMETERS by ________ Date ________ Job No. ________ Sheet No. 3

INTERRUPT VECTOR CARD #1

Columns	1–2	3–6	7–10	11–14	15–18	19–22	23–26	27–30	31–34	35–38	39–42	43–46	47–50	51–75	76–80
Field	CC	CPU #	O/S PROG. #	INTERRUPT ADDRESS: I/O TERM	R/N	FUNCT	CLOCK	RCV	TERM	END SEEK	OPEN CLOSE			INTERRUPT VECTOR CARD #1	SYST. I.D.
Entry	31			1	2	3	4	5	6	7	8	9	10		

INTERRUPT VECTOR CARD #2

Columns	1–2	3–6	7–10	11–14	15–18	19–22	23–26	27–30	31–34	35–38	39–42	43–46	47–50	51–75	76–80
Field	CC	CPU #	O/S PROG. #	INTERRUPT ADDRESS CONT'D										INTERRUPT VECTOR CARD #2	SYST. I.D.
Entry	32			11	12	13	14	15	16	17	18	19	20		

O/S MEMORY ALLOCATION CARD

Columns	1–2	3–6	7–10	11–14	15–18	19–22	23–26	27–30	31–34	35–38	39–42	43–75	76–80
Field	CC	MEMORY I.D. FOR O/S: O/S PROG #	MEM. #	O/S PROG #	MEM. #	O/S PROG #	MEM. #	O/S PROG #	MEM. #	O/S PROG #	MEM. #	O/S MEMORY ALLOCATION CARD	SYST. I.D.
Entry	34												

POISSON DEFINITION CARD #1

Columns	1–2	3–7	8–12	13–17	18–22	23–27	28–32	33–37	38–42	43–47	48–52	53–75	76–80
Field	CC	POISSON ARGUMENT										POISSON DEFINITION CARD #1	SYST. I.D.
Entry	35	0	5	10	15	20	25	30	35	40	45		

POISSON DEFINITION CARD #2

Columns	1–2	3–7	8–12	13–17	18–22	23–27	28–32	33–37	38–42	43–47	48–52	53–75	76–80
Field	CC	POISSON ARGUMENT										POISSON DEFINITION CARD #2	SYST. I.D.
Entry	36	50	55	60	65	70	75	80	85	90	95		

SECTION VI

OPERATING SYSTEM DESCRIPTIONS

6.0 WORKER ROUTINE

All W/R statements are available to the operating system.

6.1 CPU ITEM

There is a three word CPU ITEM for each CPU being represented in a simulation as explained below.

6.1.1 First Word

The first word contains the transaction word -TW (address of the worker routines transaction item - TI) of the current operating transaction -COT.

6.1.2 Second Word

The second word contains the transaction word of an available transaction - AT. A Current Operating transaction automatically becomes an available transaction upon crossing the boundary into the operating system. When this boundary is crossed, a new transaction item is produced to represent the operating system and becomes the current operating transaction.

6.1.3 Third Word

The third word contains an I/O transaction word -IOT when an I/O is to be performed.

6.1.4 Operating/Primary Statements

The statements provided for the use of the operating system and primary worker routines will deal with available or I/O transactions (CYCLE is the exception).

6.1.5 Transaction Word Manipulation

How transaction words are manipulated within a CPU ITEM.

EXAMPLE: Assume a Worker Routine goes to the operating system to perform an I/O.

WR=COT	to OS for READ
OS=COT WR=AT	upon crossing OS boundary
OS=COT WR=AT	an I/O transaction item is produced
IOTI=IOT	when OS determines and I/O is required
OS=COT WR=AT	when IOTI has been sent to Future Events Chain - FEC to represent I/O time
WR=COT	upon returning control to the worker routine to continue processing while I/O is being performed. OS transaction item is destroyed when leaving the OS boundary.

6.2 CARD FORMATS

Card formats are to include the information in accordance with the samples and the following directions.

6.3 CARD CODE

All operating system and worker routine statements must contain an 80 in columns 1-2.

6.4 WORKER ROUTINE NUMBER

All statements associated with a particular operating system or worker routine must contain that routine's number in columns 3-4.

6.5 SEQUENCE NUMBER

All operating system and worker routine statements must contain a four digit sequence number in columns 5-8. These sequence numbers must be incremented by unity for each statement.

The first worker routine statement following the routine's base statements (card type 81-85) must contain the sequence number 0001.

Operating system and worker routine base statements (card type 81-85) sequence field must contain zeros or blanks. These statements must be presented to the simulation model in the order in which they are herein described.

6.6 SYSTEM ID

Columns 76-80 may be used for a system ID. This field will not be examined by the Simulate Machine.

Columns 40-77 are not examined by the simulation model and may be used for comments.

NOTE

A. All fields (excepting 1-2 and 3-4) must be four digit fields in the form DDDE/NNNN as specified on the format sheets.

B. DDDE will be interpreted as a number in the range 1-9999 with an exponent in the range 0-9. NNNN will be interpreted as a number in the range 1-9999.

C. Unused fields may be left blank or zero filled.

D. All used fields must have their values right justified and zero filled.

E. When assigning absolute addresses to TRANSFER type statements, the address used must be relative to the first statement following the worker routine base statements (card types 81 through 85).

6.7 31 MEMORY MEM 1, MEM 2, (ZERO, QUEUE NUMBER)

NSI	NOT ENOUGH MEMORY
NSI+1	PACK NEEDED
NSI+2	MEMORY AVAILABLE

This statement examines the memory map of the object system to determine if there is sufficient storage to load an available transaction in the range of memories specified.

6.7.1 First Parameters

The first parameter specifies the first memory to be examined.

6.7.2 Second Parameter

The second parameter specifies the last memory to be examined.

6.7.3 Third Parameter

The third parameter may specify either zero or a queue number. If it is zero, the test of memory will reference the AT of the current CPU item. If it is a queue number, the test will reference the entry indicated by the current setting of the pointer for the queue specified. If a queue is specified, it must be one whose contents are designated as worker transactions. Attempting to reference a queue of I/O transactions with this statement will generate an error message and terminate the simulation.

6.7.4 Possible Exits

There are three possible exits from this statement as follows:

a. The next sequential instruction, (NSI) will be executed if there is not enough memory to run this worker routine.

b. The NSI+1 will be executed if the total storage available will satisfy the requirements of the worker routine only if the present contents of memory are compacted. The normal operation in this case would be to PACK storage.

c. The NSI+2 statement will be executed if there is enough storage to run this worker routine without packing. The normal coding at NSI+2 would be an ALLOCATE statement.

NOTE

In the memory range specified, memories not accessible to the current operating CPU will not be examined.

6.8 32 ALLOCATE MEM 1, MEM 2

The ALLOCATE statement examines the memories specified by the first operand through the second operand and assigns the quantity of memory required by the AT. If the AT is an active re-entrant program, only the required data and I/O storage will be allocated.

NOTE

A MEMORY statement should always precede the ALLOCATE statement. Attempting to perform an ALLOCATE statement when sufficient memory is not available will generate an appropriate error message and terminate the simulation.

6.9 33 DEALLOCATE

The DEALLOCATE statement examines the object memory map and releases all storage assigned to the AT subject to the following restriction. If the worker routine indicated by this TI is specified as re-entrant and there is another active TI associated with this worker routine, only data storage and I/O storage are released. For a re-entrant worker routine. instruction storage is released only when there are no active TI's in the system associated with this routine.

NOTE

The DEALLOCATE statement is normally followed by a DESTROY statement to eliminate the terminated AT from the system. Any attempt to return to RETURN with an AT which has no memory associated with it will generate an appropriate error message and terminate the simulation.

6.10 34 PACK MEM 1, MEM 2

The PACK statement causes the contents of storage to be compacted starting at the beginning of MEM 1 and continuing to the end of MEM 2. MEM 1 and MEM 2 may be the same if a single memory is to be packed.

For an operating system providing dynamic compacting of memory where necessary, the PACK statement is normally coded at the NSI+1 exit of the MEMQRY statement, followed by an ALLOCATE statement at NSI+2. For operating systems which automatically pack memory when any is released, the PACK statement is normally coded immediately following the DEALLOCATE statement.

6.11 QUEUE PROCESSING

The simulator permits the operating system to manipulate queues that are defined by the system parameters. Definition of each queue must include the following information:

A. Queue number

B. Length of queue (number of entries permitted)

C. Queueing method (program priority, FIFO, LIFO)

D. Type of entry permitted (worker routine transaction or I/O transaction).

Associated with each queue is a pointer which indicates the entry to be scrutinized when the queue is being scanned. The scanning process is accomplished by the use of the three EXAMINE statements, which do not alter the contents of the queue. Transaction words are entered on and removed from queues only by the PLACE and SELECT statements, respectively. No item may appear on more than one queue at any given time. The EXAMINE statements and the SELECT statement will ignore any queue entry which cannot be handled by the current CPU. If such an entry is the only item in a queue, the NO FIND exit will be taken.

6.12 35 EXAMINE - FIRST QUEUE NUMBER

NSI	No Find
NSI+1	Find

The EXAMINE - FIRST statement makes the first entry (if any) in the specified queue available to the operating system. This statement causes the pointer to be reset to the first entry before the queue is accessed. Contents of the queue are not altered.

This statement has two exit lines. If there is no item in the queue, the next sequential instruction (NSI) will be executed. If there is an entry in the first queue location, NSI+1 will be executed.

The O/S statement which is coded at NSI+1 depends on the nature of the queue being examined and the reason for examining it. For example, if the queue contains I/O transactions awaiting initiation which are being examined to determine if any can now be initiated, NSI+1 would logically contain and I/O READY statement.

6.13 36 EXAMINE - NEXT QUEUE NUMBER

NSI	No Find
NSI+1	Find

The EXAMINE - NEXT statement makes the next entry (if any) in the specified queue available to the operating system. This statement causes the previous setting of the queue pointer to be incremented by 1 before the queue is accessed. Contents of the queue are not altered.

This statement has two exit lines. NSI will be executed if the entry being examined is zero or the end of the queue has been reached. NSI+1 will be executed if an entry is present.

The O/S statement coded at NSI+1 follows the same principle described in connection with the EXAMINE - FIRST statement.

NOTE

This statement must be preceded by either an EXAMINE - FIRST or EXAMINE - LAST statement referencing the same queue to insure proper processing.

6.14 37 EXAMINE - LAST QUEUE NUMBER

NSI	No Find
NSI+1	Find

The EXAMINE - LAST statement makes the last entry (if any) in the specified queue available to the operating system. Contents of the queue are not altered.

This statement has two exit lines. If there is no item in the queue, the next sequential instruction (NSI) will be executed. If the queue contains one or more entries, NSI+1 will be executed.

6.15 38 PLACE (AT, IOT), QUEUE NUMBER

The PLACE statement causes the transaction word (AT or IOT) specified by the first parameter to be put on the queue specified by the second parameter according to the queueing method (priority, FIFO, LIFO) specified in the S/P queue definition card. If queueing is by priority, the current item will be placed after any other items in the queue with the same priority level. The queue pointer is set to queue start at the conclusion of execution of this statement.

NOTE

The simulator will detect as an error any attemt to place a zero item on a queue.

If the named queue is full when this statement is executed an error message will be generated and the simulation will terminate.

6.16 39 SELECT (AT, IOT), QUEUE NUMBER

The SELECT statement causes the item indicated by the pointer of the queue specified in the second parameter to be removed from the queue and placed in either the AT or IOT of the CPU item as specified in the first parameter. When the SELECT statement is interpreted, the CPU item specified

by the first parameter must be zero, as a result of either placing the item it previously contained on some queue or destroying it. If that word is not zero, an appropriate error diagnostic will be generated and the simulation aborted.

Error termination will also result from an attempt to select from a queue either a zero-item or an item which cannot run in the current CPU. Therefore, in a multi-computer system or in any case where the contents of the queue are unknown, an EXAMINE statement must precede the SELECT statement to insure that the queue contains at least one item which can run in the current CPU.

Following execution of the SELECT statement, the queue pointer is set to queue start.

NOTE

This statement will reset EXAMINE pointer.

Attempting to select an item from a queue of the wrong type (such as selecting an AT from an I/O queue) will generate an error diagnostic and terminate the simulation.

The transaction word (in the CPU item) specified by the first parameter must be zero. If it is not, an error message will be generated and the simulation will be terminated.

6.17 40 BUFF

NSI	Record Available, No I/O Necessary
NSI+1	Record Available, Initiate I/O
NSI+2	No Record Available

The BUFF statement permits the operating system to determine the necessity for performing an I/O operation based on current state of buffers. When a worker routine READ or WRITE statement is interpreted, control is transferred to the operating system at the READ/WRITE interrupt location which normally causes a BUFF statement to be executed. The BUFF statement interrogates present buffer contents for the file to be read or written to see if a. a record is available and/or an I/O operation is necessary.

The BUFF statement has three exit lines as follows:

1. NSI is executed if there is a record available and there is no buffer empty for a READ or full for a WRITE. NSI is normally coded as a RETURN to the worker program which issued the READ or WRITE command.

2. NSI+1 is executed if a record is available but one buffer is empty/full, indicating that an I/O operation should be initiated. At this point an I/O transaction item is created. Normally the NSI+1 exit should transfer control to a test for IO READY, which will either initiate the I/O operation or queue it, followed by a RETURN to the originating worker or selection of a new AT.

3. NSI+2 is executed if there are no records available. This exit can be reached only if all possible I/O operations have previously been initiated but not yet completed. The normal coding at this point may be to place the worker routine on a deferred queue and to select a new AT.

NOTE

> In those cases where a record is available, the record count for the file is updated before the NSI or NSI+1 exit is taken.

6.18 41 SEEK

NSI — Seek Initiated

NSI+1 — No Seek Required

The SEEK statement allows an operating system to perform a positioning operation on a random access device without keeping the channel and control unit busy. The seek statement first examines the device type. If the device is not a random access device, the NO SEEK REQUIRED exit is taken. The next test examines the TO-FROM Table. If the TO-FROM table entry is zero or negative, the NO SEEK REQUIRED exit is taken.

If the SEEK-INITIATED exit is taken, the IOT is placed on the Future Events Chain (FEC) and the IOT in the current CPU item is set to zero. When this item comes off the FEC, it will generate a SEEK interrupt.

If the SEEK statement is not used, the seek time specified in the TO-FROM Table will be added to the Read-Write time and will hold the channel and control unit busy for the entire operation.

6.19 42 IO READY (ZERO, QUEUE NUMBER)

NSI — BUSY

NSI+1 — READY

The IO READY statement is used to determine whether the facilities (file, device, and channel-control unit path) required for an IO operation are available. If the parameter is zero, the test is made for the current IOT. If the parameter is a queue number, the test is made for the item indicated by the current setting of the pointer for that queue.

This statement has two exit lines. NSI will be executed if any of the necessary facilities is busy, NSI+1 will be executed if all are available. If the test was made for an item on a queue, the READY exit should lead to a SELECT statement to remove that item from the queue and make it the IOT.

NOTE

If the parameter indicates a queue, it must be one whose contents are specified as I/O transactions. Attempting to perform this operation on a queue of worker transactions will generate an error message and terminate the simulation.

6.20 43 IO ADVANCE

The IO ADVANCE statement is used to indicate the initiation of an I/O operation for a worker routine. An IO ADVANCE statement must follow the READY path of an IO READY statement. The interpretation of an IO ADVANCE statement places the IOT on the future events chain (FEC) for the time required to perform the I/O operation, and clears the IOT to zero. In completion of the operation the next sequential instruction (NSI) is executed.

NOTE

A. If any file, device, control unit, or channel required for this I/O operation is busy, and not assigned to this particular worker routine, the appropriate diagnostic message will be supplied and the simulation terminated.

B. Attempting to perform an IO ADVANCE instruction with a zero IOT will generate an error message and terminate the simulation.

C. No interrupts may be allowed between an IO READY and an IO ADVANCE.

6.21 44 IO TERM QUEUE NUMBER

NSI	May not proceed
NSI+1	May proceed

The IO TERM statement tests the item indicated by the current setting of the pointer, for the queue specified, to determine if the item is a transaction item which can now proceed due to the termination of the current I/O transaction item (IOT). A transaction item is able to proceed if it is associated with the current I/O transaction item. A transaction item is not able to proceed if it is not associated with the current I/O transaction.

This statement has two exits. The next sequential instruction (NSI) is executed if the worker transaction indicated by the pointer may not proceed. The NSI+1 is executed if the worker transaction may proceed.

NOTE

The queue specified by the operand of the statement must contain a worker transaction item or else an appropriate diagnostic message will be supplied and the simulation terminated.

6.21.1 Operating System Switches

There are three operating system statements which handle the setting and testing of 200 switches provided for use by the operating systems and the primary worker routines. The three statements are SET, RESET, and TEST.

Switches numbered 101-200 are global and may be used as liaison between operating systems. Each of the 5 CPU's permitted in a simulation may have up to 20 local switches (1-20).

6.22 45 SET SWITCH NUMBER

The SET statement turns on one of the 200 operating system global switches. The switch to be turned on is indicated by the switch parameter. If the switch to be SET is already on, it will be left on.

NOTE

At the start of a simulation all switches are off.

6.23 46 RESET SWITCH NUMBER

The RESET statement turns off one of the 200 operating system global switches. The switch to be turned off is specified by the switch parameter. The RESET statement has no effect if the specified switch is already off.

NOTE

At the start of a simulation, all switches are off.

6.24 47 TEST SWITCH NUMBER

NSI SWITCH ON

NSI+1 SWITCH OFF

The TEST statement tests one of the 200 operating system global switches. The switch to be tested is indicated by the switch number parameter. There are two exits from this statement. The next sequential instruction will be executed if the switch tested is on. The NSI+1 will be executed if the switch is off.

NOTE

At the start of a simulation, all switches are off.

6.25 48 INTERRUPT CPU, INTERRUPT NUMBER

The INTERRUPT statement allows an operating system to generate an interrupt which will effectively be placed at the top of the Future Events Chain (FEC). The first operand specifies the CPU which is to be interrupted, and the scond operant specifies the number of the interrupt. The current operating program will not lose control of the CPU, unless the CPU being interrupted is the CPU executing the current operating program.

NOTE

The following conditions are recognized as errors in the use of this statement and will cause an error message and the termination of the simulation.

a. The specification of an interrupt number not defined in the operating system specified.

b. The specification of a CPU which does not use the operating system defined.

The first eight locations in the interrupt vector are reserved for pre-assigned entries into the operating system and may not be accessed by the INTERRUPT statement.

6.26 49 DISABLE (ALL, INTERRUPT NUMBER)

The DISABLE statement forces the simulator to defer the occurrence of any or all of the interrupts specified at the start of the operating system. If the parameter specified is ALL, all interrupts will be deferred. A single interrupt may be deferred by inserting its ordinal interrupt number into the INTERRUPT number parameter.

If a transaction, upon reaching the top of the Future Events Chain (FEC), finds that its interrupt action has been disabled, it will remain on the FEC until and enable statement allows it to enter the interrupt routine.

NOTE

The number 0099 indicates that ALL interrupts are to be disabled.

6.27 50 ENABLE (ALL, INTERRUPT NUMBER)

The ENABLE statement causes the simulator to stop deferring the interrupts specified by the operand. This statement is used to reverse the actions specified in a DISABLE statement. If an interrupt which was deferred by a DISABLE statement becomes active by an ENABLE statement, it will cause an interrupt as soon as a CPU advance time occurs.

NOTE

The number 0099 indicates that all interrupts are to be enabled.

6.28 51 CLOCK [illegible] (TIME IN US)

The CLOCK statement sets a simulated [illegible] a clock interrupt when it reaches zero, unless the CLOCK interrupt is disabled. When the CLOCK statement is interpreted, the value of the operand is placed into the simulated interval timer destroying any previous setting. Once the interval timer has reached zero, thereby generating an interrupt, another interrupt may not occur until the timer has been reset by a new CLOCK statement.

6.29 TIME

This field specifies in microseconds, the interval between interrupts. The value of this field must be in the form DDDE.

NOTE

In a multiple CPU environment there is one CLOCK per CPU.

6.30 53 RETURN

The RETURN statement is used to leave the operating system and return to the next statement of the routine represented by the AT. The AT will normally represent a worker routine but it may be a primary worker or another entrance in the operating system. When the RETURN statement is interpreted, the COT (representing the operating system) is destroyed and the AT (representing the worker routine) becomes the COT.

NOTE

A RETURN may not be executed if there is not AT or if there is an IOT in the current CPU item. Either of these cases results in an appropriate error message and termination of the simulation.

6.31 53 ACTIVATE PROGRAM NUMBER

The ACTIVATE statement causes the immediate generation of a transaction item for the program specified by the parameter. In addition, the transaction workd for this transaction becomes the AT in the current CPU ITEM. Before the operating system executes an ACTIVATE statement, the AT in the current CPU ITEM must be set to zero by a DESTROY or a PLACE statement.

NOTE

If it is possible for an ACTIVATE statement to be executed by a CPU which is incapable of executing the transaction [illegible] queue before it is manipulated. In this case, the SELECT statement will ignore this transaction word since the current CPU can not process it.

6.32 54 RECEIVE

The RECEIVE statement takes the transaction item associated with a program just entering the system and places it in the AT of the current CPU item. Before executing this instruction, the operating system must insure that the AT word of the current CPU item is zero by placing the AT on a queue or by destroying it.

6.33 55 CYCLE

The CYCLE statement allows an operating system to enter a wait state where the CPU is not actively processing any operating system or worker routine statements. Normally the CYCLE will be used only when there is no available transaction in the entire system. Before executing a CYCLE statement, the operating system must ensure that all interrupts are enabled, the AT is zero, and IOT is zero. If any of these conditions is not fulfilled, an error message will be generated and the simulation terminated.

6.34 56 DESTROY (AT, IOT)

The DESTROY statement causes the specified word (AT or IOT) of the current CPU item to be cleared to zero and the associated transaction item to be removed from the system.

A DESTROY statement specifying the AT is normally used only as a result of worker program termination. A DESTROY statement specifying the IOT normally follows the termination of an I/O operation.

NOTE

> If the worker routine associated with the AT being destroyed has any files that have not been closed, an error message will be generated and the simulation will terminate.

6.35 57 PERIPHERAL 0, or QUEUE NUMBER

KSI	REQUIRED FILES/DEVICES NOT AVAILABLE
KSI+1	ALL FILES/DEVICES RESERVED

The PERIPHERAL statement allows the files and devices associated with a worker routine to be examined and, if available, assigned to the worker routine.

There are two possible exits from this statement. The next sequential instruction (NSI) will be executed if the required files/devices are not available. NSI+1 will be executed if all required files/devices are available.

6.35.1 0, or Queue

If the first parameter of this statement is 0, the files associated with the AT in the CPU item will be examined for availability. If a queue number is inserted in the one parameter associated with this statement, the files associated with the transaction item indicated by the queue pointer will be examined for availability.

NOTE

The use of the PERIPHERAL statement does not obviate the need for an OPEN statement in the worker routine. The PERIPHERAL statement simply reverses files/devices. The OPEN statement actually assigns them to the worker routine if:

A. A queue is referenced, it must be the type that contains AT transaction words, or a diagnostic will be generated and the simulation will terminate.

B. A queue is referenced, and the queue pointer indicates a zero word within the queue, a diagnostic will be generated and the simulation will terminate.

6.36 INTERRUPTS

Upon the occurrence of an enabled interrupt the operating system will receive control of the CPU which was interrupted, at the location specified in the interrupt vector table for this interrupt. The following section describes the current CPU status when the operating system receives control.

6.37 #1 I/O TERMINATION

An I/O TERMINATION interrupt occurs, if enabled, when an I/O operation initiated by an IO ADVANCE statement has reached the top of the future events chain, (FEC) indicating completion. When this occurs, the current operating transaction (COT) is made the available transaction (AT), ($\overset{*}{a}$) and the I/O transaction from the top of the future events chain becomes the current I/O transaction (IOT). If there is a possibility that the transaction item, which initiated the I/O operation just terminating, was placed on a deferred queue because it could not continue, it may be found by means of the IO TERM statement.

(a) An operating system transaction item is generated as the COT.

6.38 #2 READ-WRITE

A READ-WRITE interrupt occurs when a current operating transaction executes a read or a write statement. When this occurs, the current operating transaction (COT) is made the available transaction (AT) and an operating system transaction is generated as the COT. The operating system address for a READ-WRITE interrupt is normally a BUFF statement.

NOTE

If the operating system issues a READ or a WRITE statement, the READ-WRITE interrupt must be ENABLED and the AT and the IOT must be zero. Whenever, the operating system is interrupted, any current AT will be destroyed, and an IOT may be destroyed.

6.39 #3 FUNCTION

A FUNCTION interrupt occurs when a current operating transaction executes a FUNCTION statement. When this occurs, the current operating transaction (COT) is made the available transaction (AT), and an operating system transaction is generated as the (COT), and an I/O transaction is generated and made the current IOT. The normal operating system address for a FUNCTION interrupt is an IO READY statement.

NOTE

When an operating system executes a FUNCTION statement, the function interrupt must be enabled.

6.40 #4 CLOCK

A CLOCK interrupt occurs, if enabled, when the internal timer for a CPU reaches zero. The interval time is set by the CLOCK statement, and may be used for any purpose. When the CLOCK interrupt occurs, the current operating transaction (COT) becomes the available transaction (AT) and an operating system transaction is generated as the COT.

6.41 #5 RECEIVE

A RECEIVE interrupt occurs, if enabled, when a program reaches the top of the future events chain for the first time. At this time the program is being entered into the system to compete for system facilities. When this occurs, the current operating transaction (COT) is made the available transaction (AT) and an operating system transaction is generated and made the COT. Before the transaction word identifying the program received can be made available to the operating system, the available transaction must be queued or destroyed. The new program transaction word is made the AT by the RECEIVE statement.

6.42 #6 PROGRAM TERMINATION

A PROGRAM TERMINATION interrupt occurs when any routine executes a TERM statement. When this occurs, the current operating transaction (COT) which is the terminating program is made the available transaction, and an operating system transaction is generated as the COT. It is then the operating system's function to deallocate memory and to destroy the terminating program's transaction word.

6.43 #7 END OF SEEK

An END OF SEEK interrupt occurs, if enabled, when a seek operation initiated by a SEEK statement is complete. When this occurs, the current operating transaction (COT) is made the available transaction (AT) and the seek I/O transaction word is made the IOT. At this time it is the operating system function to do a new IO READY operation to attempt to begin the actual I/O operation requested initially.

6.44 #8 OPEN-CLOSE

An OPEN-CLOSE interrupt occurs when an OPEN or CLOSE statement is executed. When this occurs, the current operating transaction (COT) is made the available transaction (AT) and an operating system transaction is generated as the COT. In addition an I/O transaction is generated as the IOT. At this point it is the operating system function to perform an IO READY operation.

6.45 99 END INPUT

The END INPUT signifies to the simulator that all input data for a given simulation has been received. It must be the last card of the input data.

INTERNATIONAL BUSINESS MACHINES CORPORATION

X24-6599-0
Printed in U.S.A.

MULTIPLE-CARD LAYOUT FORM

Company: LEO J. COHEN ASSOC.

Application: O/S STATEMENT FORMATS by ______ Date 5-9-67 Job No. CSD Sheet No. 1 of 6

MEMORY

[illegible]	W/R #	SER #	OP CODE	MEM 1	MEM 2	QUEUE #
10	NN	NNNN	0031	NNNN	NNNN	NNNN

ALLOCATE

[illegible]	W/R #	SER #	OP CODE	MEM 1	MEM 2
10	NN	NNNN	003[illegible]	NNNN	NNNN

DEALLOCATE

[illegible]	W/R #	SER #	OP CODE
30	NN	NNNN	0033

PACK

[illegible]	W/R #	SER #	OP CODE	MEM 1	MEM 2
0	NN	NNNN	0034	NNNN	NNNN

NOT REPRODUCIBLE

O/S

IBM

INTERNATIONAL BUSINES… …HINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

A24-6599-0
Printed in U.S.A.

Company: LEO J. COHEN ASSOC.

Application: O/S STATEMENT FORMATS by ______ Date 5-9-67 Job No. CSD Sheet No. 2 of 6

EXAMINE - FIRST

CARD CODE	W/K #	SEQ #	OP CODE	QUEUE #
80	NN	NNNN	0035	NNNN

EXAMINE - NEXT

CARD CODE	W/K #	SEQ #	OP CODE	QUEUE #
80	NN	NNNN	0036	NNNN

EXAMINE - LAST

CARD CODE	W/K #	SEQ #	OP CODE	QUEUE #
80	NN	NNNN	0037	NNNN

PLACE

CARD CODE	W/K #	SEQ #	OP CODE	AT/ZOT (1 = AT, 2 = ZOT)	QUEUE #
80	NN	NNNN	0038	000N	NNNN

SELECT

CARD CODE	W/K #	SEQ #	OP CODE	AT/ZOT (1 = AT, 2 = ZOT)	QUEUE #
80	NN	NNNN	0039	000N	NNNN

NOT REPRODUCIBLE

O/S

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-6599-0
Printed in U. S. A.

MULTIPLE-CARD LAYOUT FORM

Company: LEO J. COHEN ASSOC
Application: O/S STATEMENT FORMATS by ______ Date 5-9-67 Job No. CSD Sheet No. 3 of 6

BUFF

CARD CODE	W/R TR	SEQ #	OP CODE	
80	NN	NNNN	0040	
1–2	3–4	5–8	9–12	13–80

SEEK

CARD CODE	W/R #	SEQ #	OP CODE	
80	NN	NNNN	0041	
1–2	3–4	5–8	9–12	13–80

IOREADY

CARD CODE	W/R #	SEQ #	OP CODE	Ø OR QUEUE #	
80	NN	NNNN	0042	NNNN	
1–2	3–4	5–8	9–12	13–16	17–80

IOADVANCE

CARD CODE	W/R #	SEQ #	OP CODE	
80	NN	NNNN	0043	
1–2	3–4	5–8	9–12	13–80

IOTERM

CARD CODE	W/R #	SEQ #	OP CODE	QUEUE #	
80	NN	NNNN	0044	NNNN	
1–2	3–4	5–8	9–12	13–16	17–80

NOT REPRODUCIBLE

O/S

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LEO J COHEN ASSOC

Application: O/S STATEMENT FORMATS

by ______ Date: 5-9-67 Job No.: CSD Sheet No.: 4 OF 6

SET

CARD CODE	W/K #	SEQ #	OP CODE	SWITCH #	
1–2	3–4	5–8	9–12	13–16	17–80
30	NN	NNNN	0045	NNNN	

RESET

CARD CODE	W/K #	SEQ #	OP CODE	SWITCH #	
1–2	3–4	5–8	9–12	13–16	17–80
30	NN	NNNN	0046	NNNN	

TEST

CARD CODE	W/K #	SEQ #	OP CODE	SWITCH #	
1–2	3–4	5–8	9–12	13–16	17–80
30	NN	NNNN	0047	NNNN	

C/S

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U. S. A.

Company: LEO T. COHEN ASSOC.
Application: O/S STATEMENT FORMATS by ______ Date 5-9-67 Job No. CS.D Sheet No. 5 of 6

INTERRUPT

CARD CODE	W/R #	SEQ #	OP CODE	CPU #	INTERRUPT #
80	NN	NNNN	0048	NNNN	NNNN
1–2	3–4	5–8	9–12	13–16	17–20

DISABLE

CARD CODE	W/R #	SEQ #	OP CODE	INTERRUPT #
80	NN	NNNN	0049	NNNN
1–2	3–4	5–8	9–12	13–16

ENABLE

CARD CODE	W/R #	SEQ #	OP CODE	INTERRUPT #
80	NN	NNNN	0050	NNNN
1–2	3–4	5–8	9–12	13–16

CLOCK

CARD CODE	W/R #	SEQ #	OP CODE	TIME IN μS.
80	NN	NNNN	0051	DDDE
1–2	3–4	5–8	9–12	13–16

NOT REPRODUCIBLE

C/S

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LEO J. COHEN ASSOC.
Application: O/S STATEMENT FORMATS
by ________
Date: 5-9-67
Job No.: CSD
Sheet No.: 6 of 6

Card	CARD CODE (cols 1–2)	W/R # (cols 3–4)	SEQ # (cols 5–8)	OP CODE (cols 9–12)	Cols 13–16	Cols 17–80
RETURN	80	NN	NNNN	0052		
ACTIVATE	80	NN	NNNN	0053	PROGRAM # NNNN	
RECEIVE	80	NN	NNNN	0054		
CYCLE	80	NN	NNNN	0055		
DESTROY	50	NN	NNNN	0056	AT/IOT (1 = AT, 2 = IOT) 000N	
PERIPHERAL	[illegible]	[illegible]	[illegible]	[illegible]		

C/S

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company [illegible]

Application [illegible] by ______ Date ______ Job No. CSD Sheet No. ______

SECTION VII

WORKER ROUTINE STATEMENTS DESCRIPTIONS AND FORMATS

7.0 CARD FORMATS

7.1.1 Card Code

All operating system and worker routine statements must contain an 80 in Columns 1-2.

7.1.2 Worker Routine Number

All statements associated with a particular operating system or worker routine must contain that routine's number in Columns 3-4.

7.1.3 Sequence Number

All operating system and worker routine statements must contain a four digit sequence number in Columns 5-8. These sequence numbers must be incremented by unity for each statement.

The first worker routine statement following the routine's base statements (card type 81-85) must contain the sequence number 0001.

Operating system and worker routine base statements (card type 81-85) sequence field must contain zeros or blanks. These statements must be presented to the simulation model in the order in which they are herein described.

7.1.4 System ID

Columns 76-80 may be used for a system ID. This field will not be examined by the Simulate Machine.

Columns 40-75 are not examined by the simulation model and may be used for comments.

NOTE

A. All fields (excepting 1-2 and 3-4)must be four digit fields in the form DDDE/NNNN as specified on the format sheets.

B. DDDE will be interpreted as a number in the range 1-999 with an exponent in the range 0-9. NNNN will be interpreted as a number in the range 1-9999.

C. Unused fields may be left blank or zero filled.

D. All used fields must have their values right justified and zero filled.

E. When assigning absolute addresses to TRANSFER type statements, the address used must be relative to the first statement following the worker routine base statements (card types 81 through 85).

7.2 JOB

7.2.1 Card Type

Must be 81. Only one JOB statement may be associated with a Worker Routine.

7.2.2 Program Type

This field must specify one of the following:

1 = Operating System

2 = Primary Worker Routine for example, the loader routine may be considered a separate program to be called by the Operating System as required.

3 = Commercial type Worker Routine

4 = Scientific type Worker Routine

7.2.3 Load Information

This field must specify if a Worker Routine represents a non re-entrant or re-entrant type program.

1 = Non re-entrant

2 = Re-entrant

7.3 ORDINAL FILE

7.3.1 Card Type

Must be 82. There may be as many type 82 cards as required to describe the files associated with a Worker Routine.

7.3.2 Ordinal File Number

This field specifies the Ordinal File number that a Worker Routine's statements reference. For example, a READ 3 will reference the third Ordinal File statement.

7.3.3 Real File Number

This field specifies the Real File to be associated with the Ordinal File referenced in the Worker Routine's statements. If for example, W/R 3 references a common loader queue (R/F 1) as O/F 3 and W/R 4 references the same loader queue (R F 1) as O/F 5, then the connection between the Ordinal and Real files allows the Simulate Machine to determine for any Worker Routine, when it issues an I/O, the proper paths, availability and I/O time.

7.3.4 Number Buffers

This field specifies the number of buffers associated with the Real File as used by this Worker Routine.

7.3.5 Assign

This field specifies information relative to the assigning of files and devices to a Worker Routine.

1 = No assigement

2 = Assign file to Worker Routine

3 = Assign file and device to Worker Routine

NOTE

A check will be made against Configuration Data to determine if device assignment is permitted.

7.4 MEMORY 1

7.4.1 Card Type

Must be 83. Only one Memory 1 statement may be associated with a worker routine.

7.4.2 Number Instructions

This field specifies the number of instructions in the worker routine. The value of this field must be expressed in the form DDDE.

7.4.3 Number Characters

This field specifies the number of alpha characters (data) in the worker routine. The value of this field must be expressed in the form DDDE.

NOTE

The values inserted in the MEMORY 1 statement must not include INPUT-OUTPUT buffer memory requirements.

7.5 MEMORY 2

7.5.1 Card Type

Must be 84. Only one MEMORY 2 statement may be associated with a worker routine.

7.5.2 Number Numeric Constants

This field specifies the number of decimal fields in the worker routine. The value of this field must be expressed in the form DDDE.

7.5.3 Average Length (Numeric Constant)

This field specifies the average field length of the numeric constant mentioned in the previous field.

7.5.4 Number Binary Fields

This field specifies the number of fixed point (binary) fields in the worker routine. The value of this field must be expressed in the form DDDE.

7.5.5 Average Length (Binary)

This field specifies the average number of decimal digits for the decimal fields mentioned in the previous parameter.

7.5.6 Number Floating Fields

This field specifies the average number of decimal digits per floating field. The value of this field must be expressed in the form DDDE.

7.5.7 Average Length (Floating Fields)

This field specifies the average length, in decimal digits, of the previous field.

NOTE

> If the information contained in this statement is not pertinent to the simulation of a Worker Routine, the entire statement may be omitted.
>
> If any fields in a MEMORY 2 statement do not apply to a particular Worker Routine, those fields may be left blank. The memory required for INPUT-OUTPUT buffers must not be included in this statement.

7.6 GENERATE

7.6.1 Card Type

Must be 85. Only one GENERATE statement may be associated with a worker routine.

7.6.2 Spread

This field specifies (in seconds) a creation interval expressed either as a fixed number or as a Poisson function. The value of this field must be expressed in the form DDDE.

7.6.3 Fixed Poisson

This field specifies a 1 to indicate that the previous field represents a fixed number or a 2 to indicate a Poisson function.

7.6.4 Creation Limit

This field specifies the creation limit for a worker routine. If this field is left blank, the worker routine will be generated, according to the SPREAD value, until the simulation is terminated.

7.6.5 Priority

This field specifies the priority to be associated with a worker routine. The value of this field may fall within the range 0-99. If this field is zero or blank the worker routine will be assigned a standard priority as defined in the system parameter Card type 27. The highest priority will be 1 and is reserved for the Operating System Routine.

<u>NOTE</u>

Any fields that are not required may be left blank.

A [illegible] statement must be included in every worker routine and operating system routine even though all of the fields are blank.

7.7 TRANSFER

7.7.1 Card Type

Must be 01.

7.7.2 Location

This field specifies the location, within the same Worker Routine, to which an unconditional transfer is to be made.

7.8 TRANSFER PROBABILITY

7.8.1 Card Type

Must be 02.

7.8.2 Percentage

The value in this field will indicate the probability of transfer to the location named in the LOCATION field. If the probability transfer is not executed, the next sequential statement will be executed.

7.8.3 Location

This field contains the location (within the same Worker Routine) to which transfer on probability is to be made.

EXAMPLE: TRANSFER, 40, JACK

This statement indicates a 60% chance of selecting the next instruction and a 40% chance of a transfer to JACK.

7.9 READ

7.9.1 Card Type

Must be 03.

7.9.2 File Number

This field contains the Ordinal File number to be read. The Real File number associated with this Worker Routine's Ordinal File will direct the Simulate Machine to the information required to perform this input.

NOTE

When this statement is used in an operating system, the READ-WRITE interrupt must be enabled.

7.10 WRITE

7.10.1 Card Type

Must be 04.

7.10.2 File Number

This field contains the Ordinal File number to be written to.

NOTE

When this statement is used in an operating system, the READ-WRITE interrupt must be enabled.

7.11 05 FUNCTION FUNCTION NUMBER

The FUNCTION statement causes an immediate function interrupt in the operating system, and the generation of an I/O transaction item. The duration of the I/O operation initiated by the operating system as the result of the function interrupt is governed by the entries in the system function table. The function number operand of this statement specifies the entry in the system function table which is to be referenced by the IO ADVANCE. Each entry in the system function table specifies the length of time the channel, control unit, and device are to be busy with this I/O operation.

NOTE

The FUNCTION statement is not considered a normal I/O operation and should be used only for I/O operations which can not be represented in any other way. This operation may be used by worker routine.

7.12 EOF

7.12.1 Card Type

Must be 06.

7.12.2 File Number

This field specifies which ordinal file is to be checked for an EOF condition.

7.12.3 Location

This field contains the location to which a transfer is made upon detection of an EOF condition.

NOTE

This statement allows the Worker Routine to check for an EOF relative to the information given for that file in the System Parameters. The Simulate Machine keeps a tally of blocks read/written as I/O's are executed. When an EOF statement is encountered a comparison will be made between Block Count Tally and the file size (System parameters). If the Block Count Tally is greater/equal the file size, a transfer will be made to the address in the location field of the statement. Otherwise the next sequential statement will be executed.

The EOF statement may not be used to check for a file's end if that file has been opened more than once.

7.13 SUBR

7.13.1 Card Type

Must be 07.

7.13.2 Location

This field specifies the starting address (within the Worker Routine) of a subroutine.

NOTE

> A Worker Routine is permitted three nested SUBR calls at any given time. If this rule is violated, the appropriate diagnostic will be given and the simulation will terminate.

7.14 EXIT

7.14.1 Card Type

Must be 08.

NOTE

> This statement will cause the statement immediately following the last SUBR Call to be executed next.

7.15 LOOP

7.15.1 Card Type

Must be 09.

7.15.2 Value

This field specifies the number of times the transfer to the location named in the next field will occur.

7.15.3 Location

This field specifies the location (within the Worker Routine) of the next state. [illegible] Statement is executed, a LOOP Counter will be decremented and a transfer to the address in the location field will take place. When the Loop Counter is decremented to zero, the next sequential statement will be executed.

NOTE

The Worker Routine is permitted three nested LOOPS at any given time. If another LOOP statement is encountered while all three Loop Counts contain a value, it must be assumed that the logic of the Worker Routine is correct. Therefore, all Loop Counts will be set to zero and the LOOP statement just encountered will now become the first LOOP of the set.

7.16 MOVE

7.16.1 Card Type

Must be 10.

7.16.2 Number Characters

This field specifies the number of characters to be moved as a memory to memory transfer of data. A simulated time advance will be calculated by applying the logical unit size of the computer being simulated to the value of this field.

7.17 MOVE-E

7.17.1 Card Type

Must be 11.

7.17.2 Number Characters

This field specifies the number of characters to be moved within a memory on a character by character basis. The logical unit of memory of the Computer being simulated will determine the simulated time advance required to represent the action of this statement.

7.18 CO[illegible]

7.18.1 Card Type

Must be 12.

7.18.2 Number Instructions

This field specifies a value representing a mix of instructions to be executed. The Simulate Machine will, using the value of this field, produce a probability variable which will be used to calculate a simulated time advance.

7.18.3 Mark Number

This field, if non-blank, specifies the number of the Ordinal Mark Accumulator to be incremented by the value associated with the previous field. The maximum number of Ordinal Mark Accumulators that may be referenced is 10. The use of this Parameter will permit statistics to be gathered for specific sections of an Operating System or Primary Worker.

NOTE

The use of the MARK Number field is reserved for Operating Systems and Primary Workers.

If an Ordinal Mark Accumulator number greater than 10 is encountered, a diagnostic and simulation abort will occur.

7.19 MATH

7.19.1 Card Type

Must be 13.

7.19.2 Number Adds

This field specifies the number of additions (or subtractions) to be simulated.

7.19.3 Number Multiplys

[illegible]

7.19.4 Number Divides

This field specifies the number of divides to be simulated.

NOTE

The Simulate Machine will assume that the values of these fields represent separate and complete calculations. For example; in a word/register computer, one add time would include a memory to register transfer, add time and a register to memory transfer.

7.20 OPEN

7.20.1 Card Type

Must be 14.

7.20.2 File Number

This field specifies the Ordinal file number to be opened.

7.20.3 INPUT-OUTPUT

This field specifies:

1 = INPUT file

2 - OUTPUT file

7.21 CLOSE

7.21.1 Card Type

Must be 15.

7.21.2 File Number

This field specifies the ordinal file number to be closed.

7.21.3 Rewind

This field specifies if the file being closed is to be rewound or not, as follows:

1 = Rewind

2 = No Rewind

7.22 TERMINATE

7.22.1 Card Type

Must be 16.

7.22.2 Call W/R

This field is used if the immediate generation of a worker routine is dependent upon the termination of another worker routine.

BLANK/ZERO = No worker routine to be called.

NON-BLANK = The number of the worker routine to be called.

NOTE

The TERMINATE statement must be the last statement in a worker routine. If a TERMINATE statement's logical position in a worker routine is other than the last statement, it must still be placed at the end of the routine and a TRANSFER made to it from the point where it would normally have been placed.

7.23 TERMINATE STATEMENTS

It is permitted to have two TERMINATE statements, but they must be adjacent and be the last two statements in the routine. For example, if a worker routine wants to terminate and call another worker routine on a probability of 10%, the coding will appear as follows:

```
        TRANSFER     10%, JACK
        TERMINATE
JACK    TERMINATE    CALL W/R n
```

The introduction into the system of a worker routine as a result of the CALL W/R parameter will have no effect on the value of the CREATE LIMIT in the GENERATE statement.

The TERMINATE statement must appear even though it is never used. An operating system, for example, must be active until a simulation is halted by a time limit or lack of activity, but it must also have a TERMINATE statement as it's last instruction.

7.24 NO-OP

7.24.1 Card Type

Must be 99.

The NO-OP statement allows deletions to be made to a Worker Routine that has already been written in absolute format without the need to alter absolute addresses associated with TRANSFER type statements.

NOTE

The following applies to Operating System statements.

There are two ways to delete a statement that has an an NSI and an NSI+1 line associated with it.

EXAMPLE:

A.	IOREADY	=	NOP
	TRANSFER A	=	NOP
	TRANSFER B	=	NOP
B.	IOREADY	=	TRANSFER C
	TRANSFER A	=	TRANSFER A
	TRANSFER B	=	TRANSFER B
	C RETURN	=	C RETURN

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company L. D. Cohen Assoc.

Application [illegible] by ______ Date 5/1/67 Job No. CSD Sheet No. 1

CARD TYPE | NON RE-ENTRANT / RE-ENTRANT

JOB CARD

CARD TYPE | O/F | BUFF

ORDINAL FILE CARD

MEMORY 1 CARD

MEMORY 2 CARD

CARD TYPE | LIMIT

GENERATE CARD

NOT REPRODUCIBLE

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599.0
Printed in U.S.A.

Company [illegible]

Application [illegible] by ________ Date 5/1/67 Job No. CSD Sheet No. 2

SER	CARD TYPE	LOC		TRANSFER CARD
SER	CARD TYPE	C/O	LOC	TRANSFER PROBABILITY CARD

NOT REPRODUCIBLE

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company: LO CORED ASS'S.

Application: /R. STATEMENT FORMATS by ______ Date 5/1/67 Job No. CSD Sheet No. 3

Card	Fields
READ CARD	SEQ # · CARD TYPE · FILE #
WRITE CARD	SEQ # · CARD TYPE · FILE #
FUNCTION CARD	CARD TYPE · FUNCT #
EOF CARD	CARD TYPE · FILE # · LOC
SUBR. CARD	[illegible] TYPE · LOC
EXIT CARD	CARD TYPE

NOT REPRODUCIBLE

W/G

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599-0
Printed in U.S.A.

Company Leo Cohen Assoc.
Application W/R STATEMENT FORMATS by ______ Date 5/1/67 Job No. CSD Sheet No. 4

Card	Columns 1–2	3–4	5–8	9–12	13–16	17–20	21–24	25–80
LOOP CARD	20	# W/R	SEQ #	CARD TYPE	VALUE NNNN	LOC NNNN		
MOVE CARD	20	# W/R	SEQ #	CARD TYPE 10	# CHARS NNNN			
MOVE-E CARD	20	# W/R	SEQ #	CARD TYPE 11	# CHARS NNNN			
COMPUTE CARD	20	# W/R	SEQ #	CARD TYPE 12	# CHARS NNNN	MARK # NNNN		
MATH CARD	20	# W/R	SEQ #	CARD TYPE 13	# ADD. NNNN	# MULT NNNN	# DIV NNNN	
OPEN CARD	20	# W/R	SEQ #	CARD TYPE 14	FILE # NNNN	I=1 O=2 NNNN		

NOT REPRODUCIBLE

W/R

INTERNATIONAL BUSINESS MACHINES CORPORATION

MULTIPLE-CARD LAYOUT FORM

Form X24-6599
Printed in U.S.A.

Company: [illegible] Cohen Assoc.

Application: [illegible] STATEMENT FORMATS

by ________ Date 5/1/67 Job No. CSD Sheet No. 5

CARD TYPE 0025 | FILE # | CLOSE CARD

CARD TYPE 0026 | TERMINATE CARD

PRINT STATEMENT

CARD TYPE | NO - OP

NOT REPRODUCIBLE

WJR

APPENDIX A

GLOSSARY

INTRODUCTION

The explanations of the various terms contained in this glossary are directly relative to the simulation model

AT--Available Transaction

An available transaction is a transaction that is available to be manipulated by an operating system or a primary worker. It is represented, in the second word of a CPU item, by a transaction word (TW).

Buffer

A buffer is a unit of data whose dimension is expressed in characters and number of records. File length is expressed in buffers. The BUFF statement (O/S) will examine the buffer count associated with the named file when an operating system or worker routine attempts an I/O. It must be determined if an available record exists. (S/P card type 26).

C/D--Configuration Data

The configuration data is the means by which the hardware characteristics of a given computer may be entered as input data to the simulation model.

COT--Current Operating Transaction

The COT is that transaction (TW) word that occupys the first word in a CPU item and is considered to be the current operating transaction, either in a worker routine, primary worker or the operating system.

CPU Item

There is a CPU item for each CPU being represented in a simulation. A CPU item may be considered to consist of three words. The first word is used to contain a transaction work (TW) representing a current operating transaction (COT), the second word is used to contain a transaction word (TW) representing an available transaction (AT) and the third word is used to contain a transaction word representing an IO transaction (IOT). The operating system and worker routine statement descriptions specify the requirements and limitations of transaction words (TW) within a CPU item relative to a given statement.

Cycle

The cycle statement allows an operating system to enter a "wait" state where the CPU is not actively processing any operating system or worker routine statement. (O/S card type 55).

Device Penalty Time

Certain devices are capable of providing faster I/O times if the time between I/O operations does not exceed a certain limit. Device penalty time is that time which is to be added to the normal I/O time if no I/O call to a device has occurred before the specified time limit. (C/D card type 05).

Device Seizing Code

The device seizing code indicates to the simulation model whether or not a named device may be specifically assigned to one worker routine for any period of simulated time. (C/D card type 10).

Device Transfer Width

The transfer width is the number of bits which are transferred as a unit to or from a device excluding parity or other check bits. (C/D card type 05).

End of Seek

End of seek indicates to the simulation model that an I/O to a random access device has been broken into two parts. The seek and the I/O transmission. An end of seek condition causes an interrupt at the seventh word of the interrupt vector. (O/S card type 41).

EOF--End of File

EOF is a worker routine statement. (W/R card type 06).

FEC--Future Events Chain

The future events chain is used by the simulation model executive to maintain a list of all transaction items or I/O transaction items that are using simulated time. The FEC list will contain a transaction word (TW) representing a transaction item or an I/O transaction item and will contain

information that will allow the simulation model executive to determine the reason for that transaction word being on the list. The FEC is that device which allows for the possibility of continuous interrupts and is essentially responsible for incrementing the simulated clock as transaction words (TW) are removed from the future events chain and re-enter the simulated system.

FIFO

FIFO means First In - First Out and is used in conjunction with the ordering of queues.

File

A file represents INPUT/OUTPUT data on any device. Its length is determined by a file description statement (S/P card type 26) which defines the number of records per buffer, number of characters per buffer and the number per file.

Function

A function is used to represent I/O operations which cannot be represented by a normal read or write. (S/P card type 05).

Interrupt Vector

The interrupt vector is the liaison between the simulation executive and an operating system. There must be an interrupt vector associated with each CPU being represented in a simulation. The first eight addresses in the twenty location interrupt vector are pre-assigned and must contain transfers to various parts of an operating system.

IOT--INPUT/OUTPUT Transaction

An INPUT/OUTPUT transaction is generated every time a worker routine or operating system statement requiring an I/O is interpreted by the simulation executive. The IO transaction item will be maintained within the system until the IO has been completed and accounted for. An IO transaction word (IOTW) will be generated at the same time. It is this word that will be manipulated through the system as the IO transaction item remains fixed in the location assigned to it when it was generated.

IOTW--INPUT/OUTPUT Transaction Word

See IOT.

LIFO

LIFO means Last In - First Out and is used in the ordering of queues.

Load Class Entry

The load class entry statement is used to designate which of the central processor(s) being represented are able to load a given worker routine. This statement is used in conjunction with the run class statement and the program distribution statement. (S/P card type 24, S/P card type 25, S/P card type 28).

Logical Data Unit

The logical data unit is the primary unit of data processed by a given CPU. (C/D card type 01).

Memory Access Time

Memory access time is the minimum time, expressed in microseconds, between the start of a memory access and the availability of the data accessed. (C/D card type 03).

Memory Access Unit

A memory access unit is the number of usable bits, not including parity or other check bits, which are accessed by a single memory reference. (C/D card type 03).

Memory Allocation Scheme

The memory allocation scheme allows the user to describe the manner in which the simulation model is to allocate memory. (S/P card type 27).

Memory Cycle Time

Memory cycle time is the minimum time, expressed in microseconds, between the start of a memory access and the start [illegible] access to the same memory unit. (C/D card type 03).

Multiplexor Channel

A multiplexor channel is capable of [illegible] INPUT/OUTPUT message at a time. It is capable of a [illegible]

transmitting a single message (burst mode) then when transmitting multiple messages.

O/S--Operating System

At least one program designated as an operating system must be associated with each simulation. An operating system may have associated with it various primary worker routines which it will call upon to perform various operations.

Ordinal File

Ordinal files are the means by which worker routines can reference files in any named manner that is desired. Ordinal files are associated with real files by ordinal file statements (W/R card type 82).

Page

A page is a subdivision of a memory. It indicates to the simulation model, the smallest unit of memory that can be used in allocation.

Poisson Function

A poisson function is the means by which specific values can be varied. Its purpose in this simulation model is to allow the user the facility of obtaining a variable value, while preserving the mean of the distribution, in contrast to being required to use the mean value each time for those instructions that advance the simulated clock. (S/P card type 35 and card type 36).

Primary Worker

A primary worker is a subprogram of an operating system. Its priority must be lower than that of the operating system with which it is associated and it must be called by the operating system to perform its function.

Program Distribution

The program distribution statement indicates which central processor(s) may receive a worker routine and which central processor(s) may run that worker routine. The program distribution statement is used in conjunction with the load class statement and the run class statement (S/P card type 28, card type 24 and card type 25).

Real File

Worker routines are written naming ordinal files. These ordinal files are associated with real files (W/R card type 82). The real files are used when simulating the performance of I/Os for worker routines and operating systems. The real file is identified and described by an S/P card type 21.

Receive

The fifth word in the interrupt vector is considered the receive interrupt, and is the means by which an operating system is informed that a new worker routine has been generated and is ready to be entered into the simulated system.

Record

A record is that unit of data which is considered to be transferred when a worker routine or an operating system issues an I/O. It is also used to determine a buffer's size. (S/P card type 26).

Run Class Entry

Run class entry statement(s) indicates to the simulation model which central processor(s) can run a program that has been loaded into a given memory. (S/P card type 25).

Selector Channel

A selector channel differs from a multiplexor channel in that it can handle only one INPUT/OUTPUT message at a time.

S/M--Simulation Model

S/M is an abbreviation used for the simulation model in the User's Manual.

SM1

SM1 refers to the pre-simulation portion of the simulation model. It is that portion of the simulation model that accepts the input data, and checks it for accuracy and consistency. SM1 is responsible for constructing tables from the input data that will be acceptable by SM2.

SM2

SM2 refers to that portion of the simulation model which is responsible for actually running the simulation after SM1 has accepted the input data. It is also referenced as the simulation executive.

System Definition

A system definition statement presents to the simulation model such information as standard priority, memory allocation scheme and the association between operating system program(s) and CPU(s). (S/P card type 27).

S/P--System Parameters

The system parameters are the liaison between the worker routines, operating system(s) and the configuration data. Refer to the S/P portion of the User's Manual for a complete description of the input data associated with this module of the simulation model.

To-From Table

The To-From Table is constructed by SM1 and is used during the running of a simulation to determine the seek time required when initiating an I/O. It is essential y a table containing seek times between the various files that may be located on the same device.

TI--Transaction Item

A transaction item represents an active worker routine, primary worker or operating system. It is a table that contains information pertinent to the running of the program which it represents. A transaction item is generated whenever a new program enters the simulation model and will remain in the system until that program is terminated. (See TW)

TW--Transaction Word

A transaction word is associated with every transaction item in the simulation [illegible] a particular transaction item and is moved about the system as the simulation progresses. It may, for example, be considered the COT, the AT, or the IOT, in a CPU item. It is put on the future events chain when simulated time is being represented relative to a particular transaction item.

WR--Worker Routine

WR is an abbreviation found in the User's Manual. The Worker routines represent the workload being imposed on a simulated system.

APPENDIX B

DIAGNOSTICS

- System Parameters & Configuration Data Input
- Operating System & Worker Routine
- Simulation Diagnostics (SM2)
- DT Chain Dump Formats

System Parameters and Configuration Data Input

The following errors are detected by the presimulation (SM1) as it accepts the configuration data and system parameters input. These errors are serious causing a print out to the console printer and immediate termination of the simulation model.

First card not type 1

The first card in the input data is not a type 1 CPU statement. It is either missing or out of sequence.

Type 2 Card Error

A type 2 CPU statement is missing, or out of sequence.

Type 2 Card Sequence Error

The CPU ID for this statement does not match the CPU ID for a type 1 statement.

Card Type 3 Missing

There is no type 3 statement in the configuration data input or the statement is not in its proper sequence.

Number of Channel Entries Exceeds 50

The limit of 50 channels has been exceeded. A multiplexor channel is considered 2 units regarding the limit of 50.

Number of Devices Defined Exceeds 50

The restriction of a maximum 50 different types of devices has been exceeded.

Card Type 5 Sequence Error

There is no type 5 device statement or it is not in its proper sequence.

CPU Number Greater than 5

Only 5 central processors may be represented in a given simulation.

Card Type 6 Sequence Error

The CPU identification statement is missing or out of its proper sequence.

More than 10 Memories with 1 CPU

A simulation model limit has been exceeded.

Sequence Error-Card Type 7

Memory Statement information has not been received sequentially.

Card Type 7 and Type 3 Do Not Match

The memory information in the type 7 statement(s) is less than complete, or contradictory to the memory information supplied in the type 3 statement.

Card Type 7 Missing

There is no type 7 memory statement in the configuration data input, or it is out of sequence.

Channel Definition Missing

There must be a type 8 channel definition statement associated with each type 4 channel statement.

More than 20 Channels to a CPU

A simulation model limit has been exceeded.

Card Type 8 Error - CPU Number Greater Than Five

A simulation model limit has been exceeded.

Availability Table Overflow

More than 200 availability table entries have been received. Each control unit/channel path to a device is considered one entry.

Function Number Greater than 100

A simulation model limit has been exceeded. This is a result of too many functions being named by the type 22 system parameters input.

More Than 2000 Queue Entries

A simulation model limit has been exceeded. This information was received from type 23 queue statements in the system parameters.

Card Type-23 Sequence Error

The queue numbers in the type 23 card have not been received sequentially.

Queue Method Code Error

An unacceptable number has been found in the queue method field (S/P Card Type 23). The acceptable entries are as follows:

1 = FIFO

2 = LIFO

3 = Priority

Queue Type Code Error

An incorrect queue type code has been found in the queue control field (S/P Card type 23). The acceptable codes are as follows:

1 = AT

2 = IOT

Number of Queues Defined Greater than 30

A simulation model limit has been exceeded by the S/P type 23 queue statement.

Card Type 24 Sequence Error

The load class statement information concerning load class entry numbers has not been received sequentially.

More Than 15 Load Class Entries

A simulation model limit has been exceeded by the S/P type 24 load class statement.

Load Class CPU number greater than 5

A simulation model limit has been exceeded by the S/P type 24 load class statement(s).

Card Type 25 Sequence Error

The run class entry statements have not been received sequentially.

More than 5 Run Class Entries

A simulation model limit has been exceeded.

Run Class CPU Number Greater than 5

A simulation model limit has been exceeded by a type 25 system parameter statement.

Card Type 26 Sequence Error

The S/P file description statements are not sequential relative to the real file numbers.

Card Type 26 Duplicate File Number

A real file has been referenced more than once by an S/P file description statement.

File Placed on Device Number Greater than 100

The device number field of the S/P type 26 file description statement has exceeded a simulation model limit.

No Type 27 Card

The type 27 system parameter statement which is a system definition card is either missing or out of sequence.

Card Type 27 Priority Code Error

Priority Ø1 is reserved for an operating system. Standard priority for worker routines may not be less than 1 nor greater than 99.

Card Type 27 Program Number Error

No operating system, primary worker or worker or worker routine may have an ID that is less than 1 or greater than 99.

Card Type 27-No O/S Program Number

There must be at least one operating system named in an S/P type 27 system definition statement.

Card Type 27-Memory Contiquity Code Error

The acceptable code numbers that may be associated with this field in the S/P system definition card are 1 through 5. Their meaning is described in the system parameters section of the users manual (refer to S/P card type 27).

Card Type 28 Missing

An S/P Program distribution statement is required and is either missing, or out of sequence.

Card Type 28 Program Number Greater than 99

A simulation model limit has been exceeded.

Card Type 28 CPU Number Greater than 5

A simulation model limit has been exceeded.

Card Type wi Load Class Greater than 15

A simulation model limit has been exceeded.

Card Type 28 Duplicate Program Number

A redundant type 28 system parameter has been received.

No Simulation Control Card

A simulation control card is required and is either missing, or out of sequence.

Card Type 32 CPU Number Error

A simulation model limit has been exceeded by referencing a CPU number greater than 5.

Card Type 34 Error

An S/P type 34 O/S memory allocation statement is either missing, or out of sequence.

Operating System and Worker Routine (SM1)

O/S and W/R errors will cause a print out to the console printer (pause) and continue receiving input. When an error is detected in either an operating system or a worker routine, the rest of that routine will be by-passed and the next routine will be received. Serious errors will prohibit the running of SM2. If an error occurs during SM1 the dump control statement is disregarded and all dump tables are written to tape.

201

A job card has an improper card code. Field 1 must contain the number 80.

202

The first card (job card) of an operating system or worker routines does not contain an E1 in field 3.

203

A sequence number error has been found in the job card, field 3 must be zero or blank.

204

Field 2 of a job card contains a worker routine number that is less than 1.

205

Field 2 of a job card contains a worker routine number that is greater than 99.

206

Field 5 of the job card contains a program type number greater than 4, the permissable numbers are as follows:

1 = O/S
2 = P/W
3 = Commercial W/R
4 = Scientific W/R

207

Field 5 of a job card contains a program type number less than 1 (see error 206).

208

Field 6 of a job card contains a load information number that is less than 1, the permissable numbers are as follows:

1 = non-reentrant
2 = reentrant

209

Field 6 of a job card contains a load information number that is greater than 2.

210

Field 2 of a job card contains a worker routine number for which there is no program distribution card (S/P card type 28).

211

Field 2 of a job card contains a worker routine number that duplicates one already received.

212

Field 1 of an ordinal file card contains a number other than 80.

213

Field 2 of an ordinal file card contains a worker routine number that is different from the worker routine number on the job card.

214

Field 3 of an ordinal file card contains a sequence number other than 0 or blank. All W/R base statements must have a sequence number of 0 or be blank.

215

Field 4 of an ordinal file card contains a real file number greater than 150.

216

Field 1 of a memory-1 card contains a card code other than 80.

217

Field 2 of a memory-1 card contains a worker routine number that does not duplicate the worker routine number of the job card.

218

There is no memory-1 card in the worker routine base, or it is out of sequence.

219

There is no load class entry (S/P card type 24) to indicate which CPU may load this program.

220

There are no floating point times (C/D card type 02) to be used in performing arithmetic for the floating point fields defined in the MEMORY-2 statement (W/R card type 84).

221

There is no generate statement associated with this operating system or worker routine.

222

Field 2 of the generate statement contains a worker routine number that does not match that of the job card.

223

There are no fixed point times (C/D card type 02) to use when performing fixed point arithmetic for the fixed point fields defined in a MEMORY-2 statement (W/R card type 84).

224

An invalid operation code has been detected in an operating system or worker routine.

225

A statement in an operating system or a worker routine contains a worker routine number in field 2 that does not coincide with the worker routine number in the job card.

226

An operating system or worker routine statement has been read which has an improper sequence number.

227

An operating system or worker routine statement has been found that addresses a statement outside the limits of that program.

The first card following a terminate card is not a job card or an end of input card (card code 99).

Simulation (SM2)

The following is a list of diagnostics produced by SM2 as the simulation model is being run. Two types of errors are detected in this portion of the simulation model. The first type is serious, and in addition to generating an error message, will terminate the simulation. All serious errors will be identified by a secondary error number that is less that 100. The second type of error message is considered precautionary and will not stop the running of the simulation model. These error messages will have a secondary error number that is greater than 100. The error messages produced by SM2 will consist of two numbers, the first number identifies the subroutine that detected the error and the second number identifies the specific type of erroe within the sub routine. 57/1 is an example of the heading of a serious error. 14/101 is an example of a precuationary error message's heading.

The following list of SM2 errors is ordered by subroutine numbers.

Certain errors in this section will have the notation (INTERNAL) beside the error number. These errors will indicate a malfunction of the simulation model and are not correctable by the user. In the event that an INTERNAL error should occur, the user is requested to contact the simulation model designers; Leo J. Cohen Associates, 334 West State Street, Trenton, New Jersey - 609-695-1488.

7/1

An attempt has been made to transfer to a fourth nested subroutine, the limit is 3.

8/1

An EXIT statement has been encountered for which there is no return address.

13/1

A MATH statement has been encountered that has no arithmetic requirements in any of its three parameters.

14/101

The execution of an OPEN statement has opened a file that is already open.

14/1

An attempt has been made to execute an OPEN statement with an invalid I/O code. The second field of an OPEN statement must contain a 1 to signify an input file or a 2 to signify an output file.

16/1

An attempt has been made to terminate a worker routine whose available transaction (AT) in the CPU item is not zero.

31/1

In attempting to execute a MEMORY Statement, an invalid queue number has been referenced.

31/2

In attempting to execute a MEMORY statement, an invalid queue type has been referenced. The queue type must be 1, which represents an available transaction (AT) as opposed to an I/O transaction (IOT).

31/3

In attempting to execute a MEMORY statement, a memory allocation scheme as defined by an S/P card type 27 has been found to be incorrect. The value of this field must fall within the range 1-5.

32/1

An attempt has been made to execute an ALLOCATE statement without first checking if there is memory available. A MEMORY statement must be executed first.

32/2

An attempt has been made to execute an ALLOCATE statement without executing the PACK statement before attempting the allocation. This error indicates that the NSI+1 relative to the MEMORY statement did not transfer to a PACK statement.

39/1

An attempt has been made to select a zero item from a queue.

39/2

An attempt has been made to select from a queue whose type does not match the first parameter of SELECT statement. The queue type is designated by an S/P card type 23.

39/3

An attempt has been made to remove an AT or an IOT from a queue when the AT word or the IOT word in the CPU item was not empty.

39/4

An attempt has been made to associate a worker routine with a CPU that is not capable of executing that worker routine. This error is a result of executing a SELECT statement without first executing one of the EXAMINE statements.

40/101

An attempt has been made to execute a BIFF statement for a file that has not been opened.

41/1 (Internal)

IOTW error. Less than 1 or greater than 900.

41/2

An attempt has been made to execute an IOREADY statement on a queue whose type indicates that it does not contain IOTI's. The queue type is defined by an S/P card type 23.

42/3 (Internal)

The IOTW item is less than 1 or greater than 900. This error can also mean that the queue pointer is referencing a zero filled word in the queue.

42/4

An attempt has been made to execute an IOREADY statement with an I/O transaction item of zero. This is a result of not performing a BUFF statement prior to the IOREADY statement. It is the BUFF statement that indicates to the simulation model that an I/O transaction item is to be generated for an I/O.

42/5 (Internal)

Channel, control or device time is not equal to zero.

42/6 (Internal)

Channel, control of device number is not zero.

42/7 (Internal)

A real file or ordinal file number is zero.

42/8

An attempt has been made to execute an IOREADY statement where the channel number is zero. This is a result of incorrect or incomplete configuration data input on card types 9 and 10.

42/9

In attempting to execute the IOREADY statement a device number error has been found in the function table. This can be a result of no device being specified in a function entry (S/P card type 22), an invalid function number or no function definition card being entered as input data (S/P card type 22).

42/10

In attempting to execute an IOREADY statement a device number error has been found. Device numbers must fall in the range 1 to 100 as defined by configuration data card type 5.

43/1

In attempting to execute an IOADVANCE statement an IO transaction item whose number is less than 1 or greater than 900 has been encountered.

43/2

An attempt has been made to execute an IOADVANCE statement with a file number of zero.

43/3 (Internal)

IOADVANCE statement - from file equals zero.

43/4

In attempting to execute an IOADVANCE statement it has been found that the device required is still busy, this is a result of not performing an IOREADY statement first.

43/5

An attempt has been made to perform an IOADVANCE statement without first performing an IOREADY statement.

45/101 (Internal)

Switch set to zero.

45/1

A reference has been made to a switch number greater than 200.

48/101

In executing an INTERRUPT statement, reference has been made to an undefined interrupt.

48/1

In executing an INTERRUPT statement, an interrupt number greater than 20 has been found.

49/1

An attempt has been made to execute a DISABLE statement with an undefined interrupt.

52/1

An attempt has been made to execute a RETURN statement where the IOT word in the CPU item is not equal to zero.

52/2

An attempt has been made to execute an RETURN statement where the available transaction in the CPU item is equal to zero.

52/3

An attempt has been made, in executing a RETURN statement, to return to a worker routine that has not yet been loaded.

52/4

In executing a RETURN statement, an attempt has been made to connect a worker routine with a CPU that is not permitted to run it. This is a result of incorrect or incomplete data in the S/P card type 24, card type 25, or card type 28.

53/1

An attempt has been made to execute an ACTIVATE statement where the AT in the CPU item is not zero.

54/1

An attempt has been made to execute a RECEIVE statement where the AT in the CPU item is not zero.

54/2

An attempt has been made to execute a RECEIVE statement when there is no newly generated transaction item on the future events chain to be received.

55/1

An attempt has been made to execute a CYCLE statement when the AT on the CPU item is not zero.

55/2

An attempt has been made to execute a CYCLE statement when the IOT is in the CPU item is not zero.

55/3

An attempt has been made to execute a CYCLE statement without all interrupts being enabled.

56/1

An attempt has been made to execute a DESTROY statement where the first field of the DESTROY statement indicates the wrong type of transaction to be destroyed.

56/2

An attempt has been made to execute a DESTROY statement where the AT is to be destroyed and the files associated with that AT have not been closed.

57/1

An attempt has been made to execute a PERIPHERAL statement that specifies a queue with an incorrect queue control type (S/P Card type 23). Queues associated with a PERIPHERAL statement must have a queue control type of 1 (AT).

57/2

An attempt has been made to execute a PERIPHERAL statement with a queue parameter where the queue pointer references a zero filled word in the queue. This can be the result of referencing an incorrect queue, or, not having executed an EXAM statement first, to insure that the queue pointer is referencing a proper word.

57/3 (Internal)

Invalid transaction item accessed while executing a PERIPHERAL statement.

57/4 (Internal)

Invalid worder routine address encountered while executing a PERIPHERAL statement.

57/5

In attempting to execute a PERIPHERAL statement, an invalid file vector address has been encountered. This error can result from a named ordinal file not having any real file associated with it, or an undefined real file.

57/6 (Internal)

Invalid number of files encountered while executing a PERIPHERAL statement.

57/7

An attempt has been made, while executing a PERIPHERAL statement, to reference an undefined real file.

100/101

A worker routine has attempted to execute an operating system statement.

100/102 (Internal)

Non existent op-codes.

100/3 (Internal)

Internal simulation error.

103/1

The future event s chain contains more than 299 items, which is its limit. In order to correct this error, the simulation must be run with a different and/or smaller worker routine mix.

104/1

The future events chain is empty. There are no transaction items moving in the simulation model.

105/1

The future events chain is empty. There are no transaction items moving in the simulation model.

105/2

The future events chain is not empty, but no item on the chain can be moved through the simulation model. This error can result from interrupts being improperly disabled.

105/3

There is no transaction item for a given CPU on the future events chain and that CPU is not a cycle state.

110/1

An attempt has been made to receive an interrupt without the COT ord of a CPU item being zero.

110/2 (Internal)

Bad FEC code.

110/101 (Internal)

Seizing ID = ∅

110/3 (Internal)

Controller seizing ID, not IOT.

110/4 (Internal)

Device seizing ID not IOT

110/5 (Internal)

Discrepancy in real file number.

201/1

An attempt has been made to place more than 299 TI words on the future events chain. To negate this error, the simulation model must be rerun with a different and/or smaller worker routine mix.

201/2 (Internal)

Invalid code, FEC.

203/1

The simulation model, in attempting to generate a new transaction item has found that the limit of 300 active transaction items has been exceeded, to correct this error, the simulation must be rerun with a different and/or smaller worker routine mix.

204/1

The simulation model, in attempting to generate an I/O transaction item has found that the limit of 900 active IOT's has been exceeded. To correct this error, the simulation must be rerun with a different and/or smaller worker routine mix.

WORKING PAPER

APPENDIX C

STATISTICS

- Worker Routines, Final Report
- Transaction Item
- Operating System
- Queue
- Central Processor
- Memory
- Control
- Channel
- Device
- File

WORKING PAPER

INTRODUCTION

The following is a description of the statistics that may be requested by the STATISTICS CONTROL statement and the SIMULATION CONTROL statement (system parameters).

The following information is common to the description of the ten statistics tables:

1. In the header of each statistics table the number NNN refers to the interval during which that statistics table was developed. This interval is controlled by the first parameter of the SIMULATION CONTROL statement (system parameters).

2. ST number N in the header associates a given statistics table with one of the ten fields in the STATISTICS CONTROL statement (system parameters). This sytem parameter allows the user to determine which statistics tables are to be developed during a simulation.

3. For all statistics tables whose fields contain a time value, that value will be given in minutes and hundredths of minutes.

4. For all statistics tables whose fields contain a percentage value, that value will be given in percent and hundredths of a percent.

5. The statistics for a given interval will be cumulative. Therefore, the final set of statistics tables will be developed from information accumulated during the entire simulation.

WORKER ROUTINES - FINAL REPORT

This worker routine report will be produced at the end of the simulation and will be the final statistics table developed.

TOTAL W/R THRU SIMULATION

This field will specify the total number of worker routines that were completed during a simulation

W/R NUMBER

This field associates the following statistics with a particular worker routine.

TURNAROUND TIME

The three fields associated with this subheader will speficy the minimum, maximum and average turnaround time for this worker routines transaction items during a simulation.

CPU TIME

The three fields associated with this subheader will specify the minimum, maximum and average CPU utilization by this worker routines transaction items during a simulation.

NUMBER T/I GENERATED

The two fields associated with this subheader will specify the number of transaction items generated and the average for this worker routine, relative to the total number of transaction items generated during a simulation.

NUMBER IOT/I GENERATED

The three fields associated with this subheader will specify the minimum, maximum and average number of I/O transaction items generated by this worker routine's transaction items during a simulation.

NUMBER I/O OPERATIONS

The three fields associated with this subheader will specify the minimum, maximum and average number of I/O operations that were completed by this worker routines transaction items during a simulation.

T/I QUEUE TIME

The three fields associated with this subheader will specify the minimum, maximum and average number of transaction items that were queued during a simulation.

I/O QUEUE TIME

The three fields associated with this subheader will specify the minimum, maximum and average number of I/O transaction items that were queued during a simulation.

TRANSACTION ITEM

TRANSACTION ITEM NUMBER

This field identifies this line of statistics with a particular transaction item. There will be one line of statistics for each transaction item that was active to this time interval.

W/R IDENTIFICATION

This field identifies the worker routine that was responsible for the generation of this transaction item.

STATISTICAL INTERVAL

This field will indicate the statistical interval with which these transaction items are associated.

CREATE TIME

This field will specify the creation time for this transaction item.

TERMINATE TIME

This field will specify the termination time for this transaction item.

CPU TIME

This field will specify the amount of CPU Time utilized by this transaction item during its course through the simulator.

IO QUEUE TIME

This field will specify the total amount of time this tranraction item had IO transactions on queues during its existence in the simulation.

I/O OPERATIONS

This field will specify the number of I/O operations inagurated by this transaction item during its course through the simulator.

OPERATING SYSTEM

O/S NUMBER

This field associates this line of statistics with one operating system. There will be a line of statistics for each operating system represented in the simulation.

OVERHEAD TIME

This field specifies the operating systems total overhead time to this time interval.

I/O TIME

This field specifies the total I/O time associated with the operating system to this time interval.

PERCENT UTILIZATION

This field specifies the percentage of utilization by the operating system to this time interval.

NUMBER INTERRUPT

This field specifies the number of times Interrupt #1 was accessed to this time interval. There will be 20 fields to represent the 20 available interrupts. Those interrupts that have been utilized will have values associated with them.

QUEUE

QUEUE NUMBER

This field identifies one line of statistics with a particular queue. There will be a statistics line for each queue represented in the simulation.

TYPE

This field specifies what type of transaction has been [illegible] in this queue.

METHOD

This field specifies the method of insertion and removal of transaction words from this queue.

MAXIMUM ENTRIES

This field specifies the maximum number of entries in the queue.

CURRENT ENTRIES

This field specifies the current number of entries in the queue.

AVERAGE ENTRIES

This field specifies the average number of entries in the queue.

PERCENT UTILIZATION

This field specifies the percentage of utilization of the queue in entry/seconds.

CENTRAL PROCESSOR

CPU NUMBER

This field identifies a CPU with this line of statistics. There will be a line of statistics for each CPU represented in a simulation.

CYCLE TIME

This field specifies the amount of time the CPU was cycling or non-busy to this time interval.

O/S TIME

This field specifies the amount of time the CPU spent manipulating the operating system(s) to this time interval. This can be considered overhead time.

PERCENT UTILIZATION

This field specifies the percentage of time a CPU was busy to this time interval.

PERCENT OVERHEAD TIME

This field specifies the percentage of time the CPU was busy doing work for the operating system to this time interval.

MEMORY

UNIT

This field associates this line of statistics with a given memory. There will be a line of statistics for every memory represented in the simulation.

PACKS

This field will specify the total number of times a PACK statement was applied to this memory, to this time interval.

NO MEMORY

This field specifies the total number of times the NO MEMORY exit was taken from the MEMORY statement (O/S card type 31) to this time interval.

CURRENT PAGES BUSY

This field specifies the number of pages busy at this time interval.

AVERAGE PAGES BUSY

This field specifies the average number of pages associated with this memory that were busy to this time interval.

PERCENT UTILIZATION

This field specifies the percentage of utilization of this memory to this time interval in pages/seconds.

CONTROL

CONTROL NUMBER

This field identifies one line of statistics with a control. There will be a statistics line for each control represented in the simulation.

TOTAL USE TIME

This field specifies the total useage time of a control unit to this time interval.

PERCENT USE TIME

This field specifies the percentage of useage for a control unit to this time interval.

CHANNEL

CHANNEL NUMBER

This field associates a channel with this line of statistics. There will be a line of statistics for each channel represented in a simulation.

TOTAL USE TIME

This field specifies the total useage time for the channel to this time interval.

PERCENT UTILIZATION

This field specifies the percentage of time that the channel was used to this time interval.

AVERAGE TRANSFER RATE

This field specifies the average transfer rate for this channel to this time interval.

DEVICE

DEVICE NUMBER

This field identifies a device with the information that follows. There will be a line of statistics for each device represented in a simulation.

TOTAL SEEK TIME

This field specifies the total amount of seek time associated with the device to this time interval.

PERCENT SEEK TIME

This field specifies the percentage of seek time associated with the device to this time interval.

TOTAL USE TIME

This field specifies the total amount of useage time for the device to this time interval.

PERCENT USE TIME

This field specifies the percentage of time the device was used to this time interval.

FILE

REAL FILE NUMBER

This field identifies one line of statistics with the real file number. File Statistics will occur for each file represented in a simulation.

DEVICE NUMBER

This field specifies the device number with which the real file is associated.

RELATIVE LOCATION

This field specifies the relative location on the device of the real file.

TOTAL SEIZE TIME

This field specifies the total amount of time that the file was seized to this time interval.

PERCENT SEIZE TIME

This field specifies the percentage of seize time associated with the file to this time interval.

READ WRITE TIME

This field specifies the total amount of time a file was reading or writing to this time interval.

PERCENT READ WRITE TIME

This field specifies the percentage of time a file was reading or writing to this time interval.

PERCENT USE TIME SEEKING

This field specifies the percent of use time spend seeking to this time interval.

PERCENT USE PENALTY

This field specifies the percent of use time that was a result of penalty time to this time interval.

TOTAL PENALTY TIME

This field specifies the total amount of penalty time associated with this device (if any) to this time interval.

PERCENT PENALTY TIME

This field specifies the percentage of penalty time (if any) associated with this device to this time interval.

WORKING PAPER

APPENDIX D

USER'S MANUAL

for

ADP HARDWARE SIMULATOR

PREPARED FOR

DEPARTMENT OF THE ARMY

UNDER

DEFENSE SUPPLY SERVICE
CONTRACT
#DA 49-083-OSA-3306

BY

C-E-I-R, Incorporated
5272 River Road
Washington, D.C. 20016

WORKING P...

Future Events Chain Table Dump

The Future Events Chain Table Dump provides a detailed printout of each entry on the future events chain. For a detailed description of the entries found in the future events chain table, reference the description of T1.

ENTRY

The ENTRY field contains the number of the current entry in the future events chain.

CODE

The CODE field contains the code associated with the current future events chain item. All of the possible future events chain codes are provided in the description of T1.

ITEM

The ITEM field contains the number of the item which occupies this entry on the future events chain. The manner in which future events chain items are referenced is covered completely in the description of T1.

CPU Item Dump

The CPU Item Dump contains all of the information maintained during the simulation for each CPU. The arrangement of this information in the memory of the simulating computer may be found in the description of T1.

COT

The COT field contains the number of the first entry for the current operating transaction in the transaction item table dump. The current status of the COT may be determined by examining the transaction item dump.

AT

The AT field contains the number of the first entry in the transaction item table dump for the available transaction. The current status of the available transaction may be determined by examining the transaction item table dump.

IOT

The IOT field contains the number of the current IO transaction item which may be found in the IO transaction item table.

O/S PROGRAM

The O/S PROGRAM field contains the number of the worker routine which will be used as the operating system for this CPU. The number in this field is also the number of the entry in the worker routine base table for the operating system for this CPU.

MEMORIES

The column headed MEMORIES contains a list of the memories which are usable by this CPU. Although the column contains twenty entries the maximum number of memories which may be attached to any CPU is ten. The numbers contained in this column point to entries in the memory table.

CHANNELS

The column headed CHANNELS contains a list of the channels which may be accessed by this CPU.

This channel number contained in this column are internal channel numbers and may not correspond to the external channel numbers. Each of the entries in this column point to an entry in the channel table. It is important to remember that while selector channels only required one entry in the channel table, multiplexor channels required two entries in the channel table. For this reason, the channel table numbers may skip an occassional entry.

<u>INTERRUPT-ADDR</u>

The INTERRUPT ADDRESS column contains a pointer to the statement in the statement table which will be executed for each of the twenty possible interrupts for this CPU. Therefore, the first entry corresponds to interrupt number 1, the second entry corresponds to interrupt number 2, and so on.

<u>INTERRUPT-CTR</u>

The column headed INTERRUPT COUNTER contains a count of the number of times each of the twenty possible interrupts occurred during the running of this simulation. Each interrupt count corresponds to

the interrupt whose first address in the operating system is given by the entry in the interrupt address column.

CPU DEFINITION

The two columns headed CPU DEFINITION list each of the fields from the two CPU definition cards used to define this CPU. Since the field used to define a CPU may be contained within the simulator in either floating point or fixed point format, two columns are used to print these values. Fields maintained in fixed point format will be printed in the left column, and fields maintained in floating point format will be printed in the right column.

FIXED POINT TABLE

The two columns entitled FIXED POINT TABLE contain the four pairs of entries which were defined as the fixed point table in the CPU definition card.

FLOATING POINT TABLE

The two columns headed FLOATING POINT TABLE contain the three pairs of entries which were defined as the floating point table in the CPU definition card.

CLOCK ITEM TABLE

The CLOCK ITEM TABLE field contains the time in microseconds at which the next clock interrupt will occur.

TOTAL OPERATING SYSTEM TIME

The TOTAL OPERATING SYSTEM TIME field contains the total amount of time which has been charged to operating system transactions working for this CPU. This value is expressed in microseconds.

TOTAL OPERATING SYSTEM IOT QUEUE TIME

The TOTAL OPERATING SYSTEM IOT QUEUE TIME is the total amount of time IO transaction items generated by the operating system spent on queues. This field is expressed in microseconds.

TOTAL OPERATING SYSTEM TI QUEUE TIME

The TOTAL OPERATING SYSTEM TI QUEUE TIME field contains the total amount of time operating system transaction items spent on any queue in the system. This field is expressed in microseconds.

TOTAL OPERATING SYSTEM CYCLE TIME

The TOTAL OPERATING SYSTEM CYCLE TIME expresses in microseconds the total amount of time this CPU spent cycling.

GIBSON MIX TIME

The GIBSON MIX TIME field expresses the average instruction time for COMPUTE statements executed in this CPU.

Transaction Item Table Dump

The Transaction Item Table Dump contains the current status for each active transaction in the simulator. Each transaction item requires a minimum of three entries in the transaction item table. In addition, for every seven ordinal files after the first seven an additional entry will be required in the transaction item table. Each of the possible fields in a transaction item is defined below. For a complete description of the memory layout of a transaction item in the simulating computer, examine the description of T4.

NUMBER

The NUMBER field contains the number of the entry in the transaction item table which is being printed on a current line.

W/R

The W/R field contains the number of the worker routine for which the current transaction is being run. This field points to the entry in the worker routine base table.

FILE

The FILE field contains the value of the current ordinal file request for this transaction.

ID

The ID field contains the identification number for the current transaction item. This field is in the format WWNNNN, where WW is the worker routine number and NNNN is the sequence number of this transaction.

RUN

The RUN field contains a zero if the current transaction has not had memory allocated to it. If this field is not zero it contains a pointer to the run class table entry controlling the CPU's which may execute this transaction.

PRI

The PRI field contains the priority of the current transaction.

NSI

The NSI field contains the address of the next sequential instruction to be executed by this transaction in the statement table.

IOT

The IOT field contains a count of the number of IO transaction items outstanding for this transaction.

TYPE

The TYPE field contains a code which defines the current transaction's program type. In ordinal sequence, the type code defines the current transaction as: 1. An operating system. 2. A primary worker. 3. A commercial worker. 4. A scientific worker.

CPU

The CPU field contains the number of the CPU which was last used to advance time for the current transaction.

LA-1 LC-1

The LA-1 LC-1 fields contain the first loop address and first loop counter for the current transaction item.

LA-2 LC-2

The LA-2 LC-2 fields contain the second loop address and second loop counter for the current transaction item.

LA-3 LC-3

The LA-3 LC-3 fields contain the third loop address and loop counter fields for the current transaction item.

SA-1

The SA-1 field contains the subroutine exit address for the lowest level of subroutines. The SA-2 and SA-3 field contain the second and third subroutine exit addresses, respectively.

T-P

The T-P field contains the number in the transaction item table for the time entry of the current transaction.

F-P

The F-P field contains the number in the transaction item table for the first file section of the current transaction.

CREATION TIME

The CREATION TIME field contains the time expressed in microseconds at which the current transaction item was either received or activated.

TIME OFF FEC

The TIME OFF FEC field contains the time expressed in microseconds at which the current transaction is due off the future events chain if it is currently on the future events chain.

ADVANCE TIME

The ADVANCE TIME field contains the total amount of time expressed in microseconds which the current transaction must spend on the future events chain before the next sequential instruction may be executed.

TOTAL CPU TIME

The TOTAL CPU TIME field contains a total of all of the advance times executed by the current transaction. The total CPU time field is expressed in microseconds.

TI QUEUE START

The TI QUEUE START field contains the total amount of time expressed in microseconds which the current transaction item has spent on queues.

IOT TOTAL QUEUE

The IOT TOTAL QUEUE field contains the total amount of time expressed in microseconds which IO transaction items generated by this transaction item have spent on queues.

B-n

The B-n field contains the count of the number of buffers available for the ordinal file expressed by n.

R-n

The R-n field contains a count of the number of records available in the current buffer for the ordinal file expressed by n.

T-n

The T-n field contains a count of the total number of IO operations performed for the ordinal file expressed by n. If the value contained by this field is negative, the file defined is closed. If the value contained by this field is positive the file expressed by n is opened.

I/O Transaction Item Dump

The IO Transaction Item Dump gives a printout of each IO transaction item which is active at the current moment in the simulation. For a description of the actual layout of each field in an IO transaction item in the memory of the simulating computer examine the description of T4.

ITEM

The ITEM field contains the number of the current entry in the IO transaction item table.

TI NUMBER

The TI NUMBER field contains the number of the entry in the transaction item table which generated this IO transaction item.

FEC

The FEC field contains a count of the number of times a pointer to this IO transaction item appears on the future events chain.

TYPE

The TYPE field contains a code which describes the current IO transaction item type.

CPU

The CPU field contains the number of the CPU which was used to generate this IO transaction item.

CH

The CH field contains the number of the channel which is being used by this IO transaction item.

CTL

The CTL field contains the number of the control unit which is being used by this transaction item.

DEV

The DEV field contains the number of the device which is being used by this IO transaction item.

R. F.

The R. F. field contains the number of the real file for which this IO transaction item was generated.

O. F.

The O. F. field contains the number of the ordinal file for which this IO transaction item was generated.

F.#

The F.# field contains the function number, if any, for which this IO transaction item was generated.

CHANNEL END

The CHANNEL END field contains the time at which the channel being used by this IO transaction item will be released.

CONTROL END

The CONTROL END field contains the time at which the control unit being used by this IO transaction item will be realesed.

DEVICE END

The DEVICE END field contains the time at which the device being used by this IO transaction item will be released.

QUEUE START

The QUEUE START field contains the time at which the current IO transaction item was placed on any queue,if it is currently on a queue.

TOTAL QUEUE

The TOTAL QUEUE field contains the total amount of time spent by this IO transaction item on any queue in the system.

Worker Routine Base Table Dump

The Worker Routine Base Table Dump describes each of the entries being used in the worker routine base table. For a description of how each of the fields on these entries are laid out in the memory of the simulating computer examine the description of T5.

NUMBER

The NUMBER field contains the number of the current entry in the worker routine base table.

SPREAD

The SPREAD field contains the creation interval for the generation of transactions associated with this worker routine.

LIMIT

The LIMIT field defines the total number of transactions which are to be generated for this worker routine.

GEN

The GEN field contains a count of the total number of transactions generated for this worker routine.

LOAD

The LOAD field contains a pointer to the load class table which describes the CPU's capable of loading the current worker routine.

PRI

The PRI field describes the priority of any transaction created for the current worker routine.

FST INST

The FST INST field contains the number of the first statement in the statement table which will be executed by any transaction generated for the current worker routine.

CODE

The CODE field contains a code which describes whether the current worker routine is reentrant or not.

TYPE

The TYPE field describes the current worker routine type. The worker routine types in ordinal sequence are:

1. Operating system.
2. Primary worker.
3. Commercial worker.
4. Scientific worker.

NUMBER OF TI

The NUMBER OF TI field contains a count of the number of transaction items currently in memory for the current worker routine.

CPU

The CPU field contains the number of the CPU for which a RECEIVE interrupt is to be generated when a transaction item comes off the future events chain for the first time.

FILES

The FILES field contains a count of the total number of files associated with the current worker routine.

LOC

The LOC field contains a pointer to the entry in the file vector table which contains the description of the first ordinal file for the current worker routine. Ordinal files are then described in sequence in the file vector table up to the total number of ordinal files defined for the current worker routine.

INST STOR

The INST STOR field describes the total number of pages of instruction storage required for the current worker routine.

DATA STOR

The DATA STOR field describes the total number of pages required for data by the current worker routine.

I/O STOR

The I/O STOR field describes the total number of pages of I/O storage required by the current worker routine.

CALL

The CALL field contains a count of the total number of transactions generated by a CALL for the current worker routine.

File Vector Table Dump

The File Vector Table Dump contains a description of each entry in the file vector table. The file vector table contains a description of each ordinal file used by any worker routine in the current simulation. The first entry in the file vector table for the ordinal files of a specific worker routine is given by the LOC field in the worker routine base table. For a description of the actual field layouts in the memory of the simulating computer examine the description of T6.

ENTRY

The ENTRY field contains the number of the entry in the file vector table being described.

FILE

The FILE field contains the number of the real file to which the ordinal file is assigned.

BUFFERS

The BUFFERS field contains a count of the number of buffers which are to be assigned to this ordinal file for the worker routine.

C

The C field contains the seizing code specified in the ordinal file card for this file. The three possible entries in this field are seize nothing, seize the file, and seize the file and device.

Memory Table Dump

The Memory Table Dump prints a description of each entry in the memory table. There is one entry in the memory table for each memory described in the current simulation. For a description of the actual field layout in the memory of the simulating computer examine the description of T7.

MEMORY

The MEMORY field contains the number of the current memory table entry.

LOW PAGE

The LOW PAGE field contains the number in the page table of the first page contained in this memory.

HIGH PAGE

The HIGH PAGE field contains the number of the last page in the page table for this memory.

NUMBER PACKS

The NUMBER PACKS field contains a count of the number of times a pack instruction was executed for the current memory.

NUMBER NO MEMORY

The NUMBER NO MEMORY field contains a count of the number of times a no memory exit line was taken from the MEMORY statement whi referencing this memory.

PAGES BUSY

The PAGES BUSY field contains a count of the current number of pages busy in this memory.

LAST REFERENCE TIME

The LAST REFERENCE TIME field contains the last time at which the number of pages busy in this memory was altered.

MEMORY LOAD

The MEMORY LOAD field contains a total of the number of page microseconds which have been used for the current memory up to the last reference time.

Page Table Dump

The Page Table Dump prints a description of each entry in the page table. For a description of the codes used in the page table refer to the description of T8.

PAGE

The PAGE field contains the number of the page being described by the current entry.

TI

The TI field contains the number of the transaction item or worker routine base which has seized the current page if the current page is busy.

Load Class Table Dump

The Load Class Table Dump describes each entry in the load class table being used by the current simulation. For a description of the organization of the load class table in the memory of the simulating computer reference the description of T9.

CLASS

The CLASS field contains the number of the entry in the load class table being described on this line.

TI

The TI field contains the value of the entry numbered N.

Run Class Table Dump

The Run Class Table Dump describes each of the entries in the run class table described for the current simulation. For a complete description of the organization of the run class table reference the description of T10.

CPU

The CPU field contains the number of the loading CPU for which the current entry is to be used.

TI

The TI column contains the value of the entry numbered N.

Queue Table Dump

The Queue Table Dump describes the entry for each queue defined in the current simulation. For a complete description of the queue tables reference the description of T12.

QUEUE

The QUEUE field contains the number of the queue which the current entry describes.

QUEUE-START

The QUEUE-START field contains the number of the first entry in the queue entry table for the current queue.

QUEUE END

The QUEUE END field contains the number of the last entry in the queue entry table for the current queue.

QUEUE POINT

The QUEUE POINT field contains the number of the entry in the queue entry table which is currently being referenced in this queue.

ENTRY

The ENTRY field contains a code which determines whether transaction items or IO transaction items may be placed in this queue. A1 indicates the queue contains transaction items, and A2 indicates the queue contains IO transaction items.

METHOD

The METHOD field contains a code which indicates whether the queue is organized in a priority, FIFO, or LIFO method.

MAXIMUM

The MAXIMUM field contains a count which indicates the maximum number of entries which the queue contained at any time during the current simulation.

CURRENT

The CURRENT field contains a count of the current number of entries contained by this queue.

LAST REFERENCED TIME

The LAST REFERENCED TIME field contains the time at which the number of entries contained by this queue was last altered.

QUEUE LOAD

The QUEUE LOAD field contains the total number of entry-microseconds accumulated for this queue.

Queue Entry Table Dump

The Queue Entry Table Dump provides a printout of the contents of each entry for all of the queues described in the current simulation. For a detailed description of the queue entry table contents reference the description of T13.

ENTRY

The ENTRY field contains the number of the queue entry whose contents is shown under the heading ITEM.

ITEM

The ITEM field contains a printout of the value of the entry whose number is shown on the left.

File Set Table Dump

The File Set Table Dump provides a printout for each real file described in the current simulation. For a detailed description of the contents of the file set table reference the description of T14.

REAL FILE NUMBER

The REAL FILE NUMBER field contains the number of the real file being described.

SEIZING ID

The SEIZING ID field contains the number of the transaction item, if any, which has seized the current real file.

DEVICE NUMBER

The DEVICE NUMBER field contains the number of the device on which this real file resides.

REL-LOC

The REL-LOC field contains the relative location number of this real file on the device specified.

BUF-LEN

The BUF-LEN field contains the length of the buffer required to read from or write to the real

file specified.

REC

The REC field contains a count of the number of records contained in each buffer for the current real file.

SIZE

The SIZE field contains the count of the total number of blocks contained in the current real file.

LAST-TIME-SEIZED

The LAST-TIME-SEIZED field contains the time expressed in microseconds at which this real file was last seized.

TOTAL-TIME-SEIZED

The TOTAL-TIME-SEIZED field contains the total number of microseconds for which this file has been seized during the current simulation.

TOTAL-READ-WRITE-TIME

The TOTAL-READ-WRITE-TIME field contains the total number of microseconds during which the current real file was busy with read/write operations.

Device Set Table Dump

The Device Set Table Dump provides a printout for each physical device utilized by the current simulation. For a detailed description of the entries contained in the device set table, reference the description of T15.

DEV

The DEV field contains the number of the device being described by the current entry.

ID

The ID field contains the transaction item number or TO transaction item number which has seized this device.

AVAIL

The AVAIL field contains the number of the first entry in the availability table which is used to describe a channel control unit pair required to reference this device.

CLASS

The CLASS field describes the number of the entry in the device class table which will be used to

provide the physical device description for this device.

T-F

The T-F field contains the number of the first entry in the to-from table used for this device.

WIDTH

The WIDTH field contains the width of the to-from table which has been constructed for this device.

REL-LOC

The REL-LOC field contains the relative location of the last file referenced on the current device.

SEIZE-CD

The SEIZE-CD field contains the seizing code for the current device. If the seizing code is one, the device may be seized by a transaction. If the seizing code is 2 the device may not be seized by any transaction.

LAST END TIME

The LAST END TIME field contains the time at which this device finished its last IO operation.

TOTAL SEEK TIME

The TOTAL SEEK TIME field contains the total number of microseconds which have been used for seek operations on this device.

TOTAL USE TIME

The TOTAL USE TIME field contains the total number of microseconds during which this device was busy with non-seek operations.

TOTAL PEN TIME

The TOTAL PEN TIME field contains the total number of microseconds of penalty time charged to this device.

Availability Table Dump

The Availability Table Dump provides a printout of each entry contained in the availability table. For a detailed description of each entry in the availability table reference the description of T16.

ENTRY

The ENTRY field contains the number of the entry in the availability table which is being described.

PATH

The PATH field contains a code which describes the direction of the path provided by the channel control unit pair described on the right. In addition, if the path code is negative it indicates that this is the last entry in a series of availability table entries for a single device.

CHANNEL

The CHANNEL field contains the number of the channel which may be used with the control unit numbered on the right to provide a path to the device which points to this entry in the availability table dump.

CONTROL

The CONTROL field contains the number of the control unit which may be used with the channel number provided on the left to provide a path to the device which points to this entry in the availability table.

Channel Table Dump

The Channel Table Dump provides a printout of each entry contained in the channel table for the current simulation. For a detailed description of the entries in the channel table, reference the description of T17.

CHANNEL

The CHANNEL field contains the number of the channel which is being described by this entry in the channel table dump. Notice that two entries in sequence may have the same channel number. In this event, the channel being described is a multiplexor channel.

MAX-RATE

The MAX-RATE field contains the maximum transfer rate in characters per second of the current channel.

PERCENT-INTER

The PERCENT-INTER field contains the percentage interference per thousand character per second transfer rate which will be generated by the current channel.

TOTAL TIME IN USE

The TOTAL TIME IN USE field provides the total number of microseconds during which this channel was in use.

TOTAL LOAD IN CHAR

The TOTAL LOAD IN CHAR field contains the total number of characters transmitted by the current channel.

ID-OR-MPX-RATE

The ID-OR-MPX-RATE field contains the number of the seizing transaction item for a selector channel or the current transfer rate for a multiplexor channel.

Control Unit Table Dump

The Control Unit Table Dump provides a printout of each entry in the control unit table. For a detailed description of the entries in the control unit table reference the description of T18.

CONTROL

The CONTROL field contains the number of the control unit being described by this entry.

ID

The ID field contains the number of the transaction item which has seized this control unit.

TOTAL-USE-TIME

The TOTAL-USE-TIME field contains the total number of microseconds during which the current control unit was busy.

Device Class Table Dump

The Device Class Table Dump provides a printout of each entry in the device class table. For a detailed description of the individual entries in the device class table reference the description of T19.

CLASS

The CLASS field contains the number of the entry in the device class table.

TYPE

The TYPE field contains a code which describes the type of device being defined by this device class table entry.

TRANSFER-RATE

The TRANSFER-RATE field contains the transfer rate expressed in characters per second for the device being described by this entry.

WIDTH

The WIDTH field describes the transfer width to this device in bytes.

START+STOP

The START+STOP field contains the total START+STOP time required for this device expressed in microseconds.

DEVICE-TIME

The DEVICE-TIME field contains the total amount of device time expressed in microseconds required for this device.

PENALTY-LIMIT

The PENALTY-LIMIT field contains the total number of microseconds permitted between IO operations on this device before a penalty time will be invoked.

PENALTY

The PENALTY field contains the total amount of time expressed in microseconds to be added to each IO operation which requires a penalty on this device.

REWIND-MIN

The REWIND-MIN field contains the total time expressed in microseconds to rewind a tape device.

FORM-TIME

The FORM-TIME field contains the total time to skip one line if the current device is a printer.

General Simulation Table Dump

The General Simulation Table Dump provides a printout of each of the important fields contained in the general simulation table. For a detailed description of the organization of the fields in the general simulation table, reference the description of T20.

CURRENT CPU

The CURRENT CPU field contains the number of the CPU which is currently executing statements in the simulator.

PROGRAM TYPE

The PROGRAM TYPE field contains the number of the worker routine for which the current transaction is operating.

OPERATION CODE

The OPERATION CODE field contains the number of the current statement being executed by the simulator.

OPERAND-N

The OPERAND-N field contains the value of the operand whose number is N.

SUB ERROR CODE

The SUB ERROR CODE field contains the number of the error found, if any, by the subroutine whose number is given below.

NSI CODE

The NSI CODE field contains the number of instructions which are to be skipped before the next sequential instruction is accessed.

CURRENT TIME

The CURRENT TIME field contains the current value of the simulated clock contained in the simulator. The time given by this field is expressed in micro-seconds.

SUBROUTINE NUMBER

The SUBROUTINE NUMBER field contains the number of the subroutine which is currently being executed by the simulator.

FEC CODE

The FEC CODE field contains the last FEC CODE used by the PUT FEC subroutine.

FEC ITEM

The FEC ITEM field contains the number of the last item used by the PUT FEC subroutine.

STORAGE CONT

The STORAGE CONT field contains the value of the storage contiguity code entered for the current simulation.

MEM OP 1

The MEM OP 1 field contains the number of the first memory referenced by the last MEMORY statement.

MEM OP 2

The MEM OP 2 field contains the number of the second memory referenced by the last MEMORY statement executed.

LCA

The LCA field contains the address of the largest contiguous number of pages available as determined by the last MEMORY statement executed.

LCR

The LCR field contains a count of the largest number of pages of contiguous storage required to satisfy the last MEMORY statement.

SCA

The SCA field contains the address of the second largest contiguous storage area required to satisfy the last MEMORY statement.

SCR

The SCR field contains the count of the total number of pages needed to satisfy the second largest contiguous storage requirement for the last MEMORY statement executed.

NCR

The NCR field contains a count of the total number of noncontiguous pages required to satisfy the last MEMORY statement executed.

MEMORY TI

The MEMORY TI field contains the number of the transaction item for which the MEMORY statement was executed last.

PACK CODE

The PACK CODE field contains a code which determines whether a pack is required to satisfy the last MEMORY statement executed.

INTERRUPT CODE

The INTERRUPT CODE field contains the current interrupt number as determined by the FEC manipulator.

FEC MANIP TIME

The FEC MANIP TIME field contains the time at which the last item placed on the future events chain will be removed from that chain.

FEC POINTER

The FEC POINTER field contains the number of the last entry on the future events chian which was altered by the FEC manipulator.

MEMORY UNIT

The MEMORY UNIT field is utilized by SM1 and cleared before SM2 begins execution.

GEN TI PROG NUMBER

The GEN TI PROG NUMBER field contains the number of the transaction last generated by the generate transaction subroutine.

GEN IOTI TI NUMBER

The GEN IOTI TI NUMBER field contains the number of the last IO transaction item generated.

STANDARD PRI

The STANDARD PRI field contains the standard priority which is to be assigned to any worker routine which does not have its own priority.

STAT INTERVAL

The STAT INTERVAL field contains the time, expressed in microseconds, between statistical intervals.

NEXT STAT TIME

The NEXT STAT TIME field contains the time, expressed in microseconds, at which statistics will next be printed out.

TOTAL NUMBER STATS

The TOTAL NUMBER STATS field contains a count of the total number of statistical intervals to be printed.

NUMBER STATS DONE

The NUMBER STATS DONE field contains a count of the total number of statistical intervals which have been completed.

TI ADDR

The TI ADDR is not used in the current simulator.

INTERFERENCE

The INTERFERENCE field contains the current interference rate existing in the simulator.

INT START TIME

The INT START TIME field contains the time, expressed in microseconds, at which the current interference rate began.

INT SEC

The INT SEC field contains the total number of interference microseconds which have been accumulated in the simulator.

QUEUE LIMIT

The QUEUE LIMIT field contains the address of the last entry in a queue entry table for the queue which is currently being manipulated.

QUEUE INCREMENT

The QUEUE INCREMENT field contains the number which will be added to the queue pointer to get the next entry in the current queue.

RANDOM NUMBER

The RANDOM NUMBER field contains the last random number generated by the simulator.

RANDOM NUMBER SEED

The RANDOM NUMBER SEED field contains the number to which the random number generator was initialized.

INT RANDOM NUMBER

The INT RANDOM NUMBER field contains the binary random number expressed as an integer value.

Function Table Dump

The Function Table Dump provides a printout of each entry contained in the function table. For a detailed description of the function table entries reference the description of T21.

FUNCTION

The FUNCTION field contains the number of the current entry in the function table.

CHANNEL-TIME

The CHANNEL-TIME field contains the time, expressed in microseconds, for which the channel is to be busy when this function number is used by a function statement.

CONTROL-TIME

The CONTROL-TIME field contains the time, expressed in microseconds, for which the control unit is to be busy when this function number is referenced by a function statement.

DEVICE-TIME

The DEVICE-TIME field contains the time, expressed in microseconds, for which the device is to be busy when this function entry is referenced by a function statement.

DEV

The DEV field contains the number of the device to which this function is to be directed.

Statement Table Dump

The Statement Table Dump provides a printout of each entry in the statement table. For a detailed description of the statement table entries reference the description of T22.

Since each operation code and operand entered into the statement table takes one location all of the values contained in the statement table are printed out in a large rectangular array. The statement table dump is read by looking down the number column until the address desired rounded down to the nearest unit of ten is found. The proper entry is then found by looking across that row until the proper unit value is found at the heading of the column.

Switch Table Dump

The Switch Table Dump provides a printout of all of the current values in the switch table. For a detailed description of the entries found in the switch table, reference the description of T23.

The local switches numbered 1 to 20 for CPU number 1 will be found in the switch table as entries 1 to 20. The local switches numbered 1 to 20 for CPU number 2 will be found in the switch table as numbered from 21 to 40. The local switches numbered 1 to 20 for CPU number 3 will be found in the switch table as numbered from 41 to 60. The local switches numbered from 1 to 20 for CPU number 4 will be found in the switch table as numbered from 61 to 80. The local switches numbered from 1 to 20 for CPU number 5 will be found in the switch table as numbered from 81 to 100. The global switches available to all CPU's numbered from 101 to 200 will be found under the same numbers in the switch table dump.

The switch table dump is referenced in the same way as the statement table dump.

To-From Table Dump

The TO-FROM Table Dump provides a printout of the current values for each entry in the TO-FROM table. For a detailed description of the entries found in the TO-FROM table reference the description of T24.

The TO-FROM table is referenced in the same manner as the statements table and switch table.

Mark Time Table Dump

The Mark Time Table Dump provides a detailed printout of the current values of all of the mark time accumulators in the current simulation. The mark number accumulators are numbered ordinally as indicated on the printout and the time accumulated in each of the mark time accumulators is expressed to the right in micro-seconds.

SNAP Dump

The SNAP Dump provides a more detailed printout of the current simulator status than the TRACE Dump. A SNAP Dump is printed each time there is a change in the current operating CPU, current operating transaction, available transaction, or I/O transaction item. The information used by the SNAP Dump is obtained from the tables maintained by the simulator.

SNAP NUMBER

The SNAP NUMBER field contains the number of the current SNAP Dump.

CPU

The CPU field contains the number of the current operating CPU.

COT

The COT field contains the number of the current operating transaction.

AT

The AT field contains the number of the current available transaction.

IOT

The IOT field contains the number of the current I/O transaction item.

TIME

The TIME field contains the current simulated time.

FEC-CD

The FEC-CD field contains the current FEC CODE from the general simulation table.

ITEM

The ITEM field contains the current FEC ITEM from the general simulation table.

SUB

The SUB field contains the current subroutine number from the general simulation table.

INT

The INT field contains the current interrupt number from the genrral simulation table.

NSI

The NSI field contains the current next sequential instruction counter from the general simulation table.

COT

The line which begins with the COT field describes the current operating transaction.

W/R

The W/R field contains the number of the worker routine for which the current operating transaction is being executed.

OFN

The OFN field contains the current ordinal file request for the current operating transaction.

ID

The ID field contains the identification number for the current operating transaction.

R-C

The R-C field contains a pointer to the run class for the current operating transaction.

PRI

The PRI field contains the priority of the current operating transaction.

NSI

The NSI field contains the next sequential instruction pointer for the COT.

IOT

The IOT field contains a count of the number of I/O transaction items outstanding for the COT.

TYPE

The TYPE field contains the program type for the COT.

LA-n LC-n

The LA-n LC-n fields contain the loop address and loop counter respectively for the loop whose level is n.

S-n

The S-n field contains the next sequential instruction counter for the subroutine whose level is n.

P-L

The P-L field contains the number of spaces required for the last PRINT statement exectued.

T-P

The T-P field contains the address of the time section of the current operating transaction.

F-P

The F-P field contains the address of the file section for the COT.

FEC

The FEC field contains the number of the CPU which last put the COT on the future events chain.

CREATION

The CREATION field contains the time at which the current transaction was created.

TIME-OFF-FEC

The TIME-OFF-FEC field contains the time at which the current transaction last came off the future events chain.

ADV-TIME

The ADV-TIME field contains the time which the current transaction must spend on the future events chain before it may exectue the next statement.

CPU-TIME

The CPU-TIME field contains the total amount of CPU time utilized by the current transaction.

Q-SRT-TIME

The Q-SRT-TIME field contains the time at which the current transaction item was placed on a queue if it is currently on a queue.

TI-QUEUE-T

The TI-QUEUE-T field contains the total amount of time which the current operating transaction has spent on all queues.

IOT-QUEUE-T

The IOT-QUEUE-T field contains the total amount of time which I/O transaction items associated with the current operating transaction has spent on queues.

B-n

The B-n field contains the number of buffers available for the ordinal file numbered n.

R-n

The R-n field contains the total number of records available for the ordinal file numbered n.

T-n

The T-n field contains the total number of IO operations performed for the ordinal file numbered n.

TRACE Dump

The TRACE Dump provides a single line of information about the current status within the simulator each time a new subroutine is executed. This information is extracted from a number of tables maintained by the simulator.

NUMBER

The NUMBER field provides a line count for the TRACE subroutine.

CPU

The CPU field contains the number of the current operating CPU.

COT

The COT field contains the number of the current operating transaction for the current operating CPU.

AT

The AT field contains the number of the available transaction it any for the current CPU.

IOT

The IOT field contains the number of the IO transaction for the current CPU.

OP

The OP field contains the current operation code for the current operating CPU.

F-n

The F-n field contains the operand whose number is n for the current operating CPU.

TIME

The TIME field contains the current simulated time.

F-CD

The F-CD field contains the FEC CODE from the general simulation table.

ITEM

The ITEM field contains the FEC ITEM field from the general simulation table.

SUB

The SUB field contains the number of the subroutine which is currently being executed.

INT

The INT field contains the current interrupt number from the general simulation table.

COT-NSI

The COT-NSI field contains the next sequential instruction pointer for the current operating transaction.

AT-NSI

The AT-NSI field contains the next sequential instruction pointer for the available transaction if any.

NSI

The NSI field contains the next sequential instruction counter from the general simulation table. This field indicates how many instructions are to be skipped before the next operation code is accessed.

ERR

The ERR field contains the error code from the general simulation table.

PROG

The PROG field contains the number of the worker routine which is currently being executed.

RAND

The RAND field contains the current random number from the general simulation table.

IOT

The IOT field informs the user that the line of information which follows describes the current IO transaction item if any.

TI

The TI field contains the number of the transaction item which generated this I/O transaction item.

FEC

The FEC field provides a count of the number of times a pointer to this I/O transaction item appears on the future events chain.

TYPE

The TYPE field contains a code which describes the type of I/O transaction item.

CPU

The CPU field contains the number of the CPU which created this I/O transaction item.

CHAN

The CHAN field contains the number of the channel which is being used by this I/O transaction item.

CPL

The CPL field contains the number of the control unit being used by this I/O transaction item.

DEV

The DEV field contains the number of the device being used by this I/O transaction item.

RFN

The RFN field contains the number of the real file for which this I/O operation is being performed.

OFN

The OFN field contains the ordinal file number for which this I/O transaction item is being performed.

F-NUMBER

The F-NUMBER field contains the function number which was used to generate this I/O transaction item, if any.

CHANNEL-END

The CHANNEL-END field contains the time at which the channel being used by this I/O operation will be released.

CONTROL-END

The CONTROL-END field contains the time at which the control unit being used for this I/O operation will be released.

DEVICE-END

The DEVICE-END field contains the time at which the device being used by this I/O operation will be released.

Q-START

The Q-START field contains the time at which this I/O transaction item was placed on a queue if it is currently on a queue.

Q-IOT

The Q-IOT field contains the total amount of time this I/O transaction item has spent on queues.

FILE

The FILE field informs the user that the following line of information describes the file being used by the I/O transaction item, if any.

ID

The ID field contains the number of the transaction which has seized this file.

DEV

The DEV field contains the number of the device on which this file resides.

LOC

The LOC field contains the relative location of this file on the device described.

BUFF

The BUFF field contains the length of the buffer required to READ from or WRITE to this file.

REC

The REC field contains a count of the number of records contained in a single buffer on this file.

SIZE

The SIZE field contains the total number of blocks contained in this file.

TIME-SEIZED

The TIME-SEIZED field contains the time at which this file was last seized.

T-TIME-SEIZED

The T-TIME-SEIZED field contains the total amount of time during which this file has been seized.

R/W TIME

The R/W TIME field contains the total amount of time during which this file was busy with READ or WRITE operations.

DEV

The DEV field informs the user that the line which follows describes the device being used by the current I/O transaction item, if any.

ID

The ID field contains the number of the transaction which has seized this device.

AVAIL

The AVAIL field contains the address of the availability table entry for this device.

CLASS

The CLASS field contains the address of the device class describing this physical device.

T-F

The T-F field contains the address of the to-from table, if any, used for this device.

SIZE

The SIZE field contains the size of each entry in the to-from table described previously.

L-F

The L-F field contains the number of the last file referenced on this device.

CD

The CD field contains a code which determines whether the current device may be seized or not.

LAST-END-TIME

The LAST-END-TIME field contains the time at which this device was last freed.

TOT-SEEK-TIME

The TOT-SEEK-TIME field contains the total amount of time seeking on the current device.

TOT-USE-TIME

The TOT-USE-TIME field contains the total amount of time used on this device.

PENALTY-TIME

The PENALTY-TIME field contains the total amount of time performing penalty operations used on this device.

CHAN

The CHAN field informs the user that the following line describes the channel being utilized by the I/O transaction item, if any.

ID

The ID field contains the transaction number of the transaction currently using this channel.

RATE

The RATE field contains the maximum transfer rate available on this channel.

INTER

The INTER field contains the percentage interference per thousand characters per second transfer rate for the current channel.

USE-TIME

The USE-TIME field contains the total amount of time the current channel has been used.

LOAD

The LOAD field contains the total number of characters transferred over this channel.

MPX-RATE

The MPX-RATE field contains the current transfer rate if the channel is being used in multiplexor mode.

MAX-RATE

The MAX-RATE field contains the maximum permissable transfer rate for this channel when operating in multiplexor mode.

INTER

The INTER field contains the interference rate for the current channel when used in multiplexor mode.

LOAD

The LOAD field contains the number of characters transmitted across this channel in multiplexor mode.

DT1

EVENTS CHAIN

DUMP

FUTURE EVENTS CHAIN DUMP # - NNNN

ENTRY	CODE	ITEM	ENTRY	CODE
1	XXXX	XXXX	51	XXXX
2	XXXX	XXXX	52	XXXX
3	XXXX	XXXX	53	
4			54	
5			55	
6			56	
7			57	
8			58	
9			59	
10			60	

A

[illegible]

ITEM	ENTRY	CODE	ITEM	ENTRY	CODE	ITEM
XXX	51	XXXX	XXXX	101	XXXX	XXXX
XXX	52	XXXX	XXXX	102	XXXX	XXXX
XXX	53			103		
	54			104		
	55			105		
	56			106		
	57			107		
	58			108		
	59			109		
	60			110		

B

* DTR *

CPU ITEM DUMP

CPU ITEM FOR CPU # 1 DUMP # NNNN

CPT = XXXX RT = XXXX ICT = XXXX O/S [illegible] =

[illegible]ES	CHANNELS	INTERRUPT [illegible]	[illegible]
[illegible]	XX	[illegible]	
[illegible]	XX	[illegible]	
[illegible]	XX	[illegible]	
[illegible]	XX	[illegible]	
[illegible]	XX	XXXXX	
[illegible]	XX	XXXXX	
[illegible]	[illegible]	XXXXX	
[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	XXXXX	

FIXED POINT TABLE

XX	XX
XX	XX
XX	XX
XX	XX

FLOATING POINT TABLE

XX	XX
XX	XX
XX	XX

A

[illegible]

[illegible] O/S PROGRAM [illegible]

INTERRUPT ADDR [illegible] | INTERRUPT CTR | O/S CONDITION

[illegible]

FLOATING POINT TABLE

XX XX
XX XX
XX XX

B

* DT3 *

TRANSACTION ITEM

DUMP

```
10          TRANSACTION ITEM DUMP * [illegible]

20          I  [illegible] FILE  [illegible]  [illegible]  PRI  [illegible]  IST  TYPE  CPU  LA
30          XXX  XX  XXX  [illegible]  XX  XX  XXXXXX  XX  X  XX  XXX

40          T  CREATION TIME  [illegible]  ADVANCE TIME
50          XXX  XXXXXXXXXX  XXXXX  [illegible]  [illegible]

60          [illegible]  [illegible]-1  [illegible]-1  T-1  [illegible]-1  [illegible]  T-[illegible]  [illegible]
70          [illegible]  XXXX  XXXXX  XXXXXXX  XXXX  XXXXX  [illegible]  XXXX

80          [illegible]  [illegible]  [illegible]  T-[illegible]  [illegible]-7  [illegible]-7  T-7  [illegible]

[illegible]  [illegible]  [illegible]-1[illegible]  [illegible]-1[illegible]  T-1[illegible]  [illegible]-12  [illegible]  T-12  [illegible]-13
```

A

[illegible]

[illegible] TYPE [illegible] LC-1 LC-2 [illegible]
[illegible] XX [illegible] XX X XX XXXXX [illegible]

[illegible] ADVANCE TIME TOTAL CPU TIME [illegible]

T-1 C-1 R-1 T-2 C-2 R-2 T-3 [illegible]
[illegible]

T-6 C-7 [illegible] T-7 C-7 [illegible] T-8 [illegible]

T-11 C-11 [illegible] T-12 C-12 [illegible]

* DTH *

I/O TRANSACTION

ITEM DUMP

I/O TRANSACTION ITEMS DUMP FORMAT

ITEM	TIME	FEC	TYPE	DEV	CH	CYL	DIV	K.F.	D.F.	F.
XXX	XXXX	XX	XXX	XX	XX	XX	XX	XXX	XXX	X

A

ITEMS [illegible]

[illegible]	[illegible]	[illegible]	DIV	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	XX	XX	XX	XXX	XXX	XXX	XXXXX	[illegible]	[illegible]

NOT REPRODUCI[illegible]

13

DTS

WORKER ROUTINE

BASE DUMP

WORKER ROUTINE BASE DUMP # NNNN

#	SPREAD	LIMIT	GEN	LOAD	PRI	FST INST
XXX	. .XXXXEXX	XXXXX	XXXXXX	XX	XX	XXXXX

A

BASE DUMP # NNNN

	GEN	LOAD	PRI	FST INST	CODE	TYPE	# TT	CPU	FILES	REC	[illegible]	[illegible]
	XXXXXX	XX	XX	XXXXX	X	X	XXXX	X	XX	XX	XXX.XXX	[illegible]

B

DT6

ILE VECTOR TABLE

DUMP

FILE VECTOR TABLE DUMP # NNNN

ENTRY	FILE	BUFFERS	C	ENTRY	FILE	BUFFERS	C
XXX	XXXX	XXXXXXX	X	XXX	XXXX	XXXXXXX	X

A

DUMP # NNNN

ENTRY	FILE	BUFFERS	C	ENTRY	FILE	BUFFERS	C	ENTRY	FILE
XXX	XXXX	XXXXXXX	X	XXX	XXXX	XXXXXXX	X	XXX	XXXX

B

DT7

MEMORY TABLE DUMP

MEMORY TABLE DUMP # UNNN

MEMORY	LOW-PAGE	HIGH-PAGE	# PACKS	#
XX	XXXX	XXXX	XXX,XXXX	

A

NNNN

HIGH-PAGE	# PACKS	# IN HISTORY	FREE-[illegible]	LAST-[illegible]
XXXX	XXXXXXX	XXXXXXX	XXXX	XXX [illegible]

B

*DT8 *

PAGE TABLE DUMP

```
PAGE TABLE DUMP # NNNN

PAGE        TI              PAGE        TI
XXXX        XXX             XXXX        XXX
```

A

XXX

* DT9 *

[illegible] CLASS TABLE

DUMP

A

[illegible] TABLE DUMP [illegible]

B

* DT1R *

.01 TABLE

DUMP

A

NOT REPRODUCIBLE

B

DT13

UEUE ENTRY TABLE

DUMP

QUEUE ENTRY TABLE DUMP

A

E DUMP

NOT R

B

* DT14 *

ILE SET TABLE

DUMP

FILE SET TABLE DUMP # NNNN

REAL-FILE-#	[illegible]-ID	DEVICE #	REL-LOC	BUF-LEN
XXX	XXXXX	XXX	XXXX	XXXXX
XXX	XXXXX	XXX	XXXX	XXXXX

A

DUMP # NNNN

```
ID   DEVICE #   REL-LOC   BUF-LEN   FILE-SIZE   LAST-TIME-SEIZED   [illegible]
        XXX       XXXX     XXXXX    [illegible]
        XXX       XXXX     XXXXXX   XXXXXXXX
```

NOT REPRODUCIBLE

B

* DT15 *

EVICT SET TABLE

DUMP

[illegible] TABLE DUMP # [illegible]

[illegible]	[illegible]	[illegible]	[illegible]	T-F	[illegible]	[illegible]	SE
XXX	XXXX	XXX	XXX	XXX	XXX	XX.	

A

[illegible] # [illegible]

S	T-F	WIDTH	[illegible]-LOC	SE[illegible]-CD	[illegible]	[illegible]	[illegible]
	XXX	XXX	XX.	X	[illegible]	[illegible]	[illegible]

B

DT16

AVAILABILITY TABLE

DUMP

AVAILABILITY TABLE DUMP # [illegible]

ENTRY	PATH	CHANNEL	CONTROL
XXXX	XX	XXX	XXX
XXXX	XX	XXX	XXX

A

[illegible]

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
XXX	XXX	XXX	[illegible]	[illegible]	[illegible]
XXX	XXX	XXX	[illegible]	[illegible]	[illegible]

b

A

B

DT 18

NTROL UNIT TABLE
DUMP

CONTROL UNIT TABLE DUMP # NNNN

CONTROL	ID	TOTAL USE-TIME
XXXX	XXXXX	XXXXXXXXXXXXXXXXX

A

DUMP # NNNN

TOTAL-USE-TIME	CONTROL	ID	TOTAL-USE-TIME
XXXXXXXXXXXXXXX	XXXX	XXXXX	XXXXXXXXXXXXXXXXX

B

* DT19 *

VICE CLASS TABLE
DUMP

DEVICE CLASS TABLE DUMP. # NNNN

CLASS	TYPE	TRANSFER-RATE	WIDTH	[illegible]
XXX	XX	XXXXXXX	XX	XXXXX

A

DUMP. # NNNN

R-RATE	WIDTH	START-STOP	DEVICE-TIME		
XXXX	XX	XXXXXXX.	[illegible]	[illegible]	[illegible]

B

10	GENERAL SIMULATION TABLE DUMP # NNNN		
20	CURRENT CPU	=	LCA
30	PROGRAM TYPE	=	LCR
40	OPERATION CODE	=	SCA
50	OPERAND 1	=	SCR
60	2	=	NCR
70	3	=	MEMORY TI
80	4	=	PACK CODE
90	5	=	INTERRUPT
100	SUB ERROR CD	=	FEC MANIP
110	NSI CODE	=	FEC POINT
120	CURRENT TIME	=	MEMORY UN
130	SUBROUTINE NO.	=	GEN TI PR
140	FEC CODE	=	GEN IOT
150	FEC ITEM	=	STANDARD
160	STORAGE CONT	=	STAT. INT
170	MEM OP 1	=	NEXT STAT
180	MEM OP 2	=	TOTAL #

A

TABLE DUMP # NNNN

LCA	=	# STATS DONE	=
LCR	=	TI ADDR STAT	=
SCA	=	[illegible]	=
SCR	=	INT START TIME	=
NCR	=	INT SEC	=
MEMORY TI	=	Q LIMIT	=
PACK CODE	=	Q INCREMENT	=
INTERRUPT CODE	=	RANDOM #	=
FEC MANIP TIME	=	RANDOM # SEED	=
FEC POINTER	=	INT RANDOM #	=
MEMORY UNIT	=		
GEN TI PROG #	=		
GEN IOTI TI #	=		
STANDARD PRIORITY	=		
STAT. INTERVAL	=		
NEXT STAT TIME	=		
TOTAL # STATS	=		

B

DT21
UNCTION TABLE
DUMP

FUNCTION TABLE DUMP # NNNN

FUNCTION	CHANNEL-TIME	CONTROL-TIME	DEVICE
XXX	XXXXXXXX.	XXXXXXXX.	.XXXXX

A

CONTROL-TIME	DEVICE-TIME	DEV	FUNCTION	[illegible] TIME	[illegible]
XXXXXXXX.	.XXXXXXXX.	XXX	XXX	[illegible]	[illegible]

B

DTA2

STATEMENT TABLE

DUMP

STATEMENT TABLE DUMP T [illegible]

NUMBER	1	2	3	4
0	XXXXXXX	XXXXXXX	XXXXXXX	XXXXXXX
10				
20				
30				
40				
50				

A

4 5 6 7

XXXX XXXXXXX XXXXXX XXXXXXX XXXXXXX

B

DT23

ITCH TABLE

DUMP

SWITCH TABLE DUMP # NNNN

NUMBER	1	2	3	4
00	XX	XX	XX	XX
10	XX	XX	XX	XX
20	XX	XX	XX	XX
30				
40				
50				
60				
70				
80				
90				
100				
110				
120				
130				
140				
150				
160				

A

3	4	5	6	7	[illegible]	[illegible]	[illegible]
xx	xx	xx	x	xx	xx	xx	
xx	xx	xx	xx	xx	xx	xx	xx
xx	xx	xx	xx	xx	xx	xx	

NOT REPRODUCIBLE

b

DT24

TO-FROM TABLE DUMP

TO-FROM TABLE DUMP # NNNN

ENTRY	T-1	T-2	T-3	T-4
00	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX
10				
20				
30				
40				
50				
60				
70				
80				
90				
100				
110				
120				
130				
140				
150				

A

T-3	T-4	T-5	T-6	T-7	T-8	T-9	T-10
XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX

b

TRACE

TRACE DUMP

#	CPU	COT	AT	IOT	OP	F-1	F-2	F-
XXXXX	X	XXX	XXX	XXX	XXX	XXXXX	XXXXX	XXX
XXXXX	X	XXX	XXX	XXX	XXX	XXXXX	XXXXX	XXX

A

OP	F-1	F-2	F-3	F-4	TIME	F-CD	ITEM	SUB	INT	OUT-MSG	IN-MSG
XXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXXXXXXXX	XXX	XXXX	XXX	XXX	XXXXXX	XXXXX
XXX	XXXXX	XXXXX	XXXXX	XXXXX	XXXXXXXXXXX	XXXX	XXXX	XXX	XXX	XXXXXX	XXXXX

B

TRANSACTION TERMINATION LIST

INTERVAL	TI - ID	TERM TIME	CREATION
XXX	XXXXXXX	XXXXXXX.XXX	XXXXXX

FILE #	I/O OPERATIONS	FILE #
XX	XXXXXXXXXX	XX
XX	XXXXXXXXXX	XX
XX	XXXXXXXXXX	XX
XX		
XX		
XX		
XX		

A

ST

TIME	CREATION TIME	TOTAL CPU TIME	TI QUEUE TIME
X.XXX	XXXXXXX.XXX	XXXXXXX.XXX	XXXXXXX.XXX

FILE #	I/O OPERATIONS	FILE #	I/O OPERATIONS
XX	XXXXXXXXXX	XX	
XX	XXXXXXXXXX	XX	
XX	XXXXXXXXXX		

b

W/R STATISTICS.

WORKER ROUTINE STATISTICS W/R # [illegible] NOT USED

NUMBER OF TRANSACTIONS GENERATED = NNNN

	MINIMUM	
TURNAROUND TIME	XXXXXXX.	MIN.
CPU TIME	XXXXXXX	MIN.
T/I QUEUE TIME	XXXXXXX	MIN.
IOT QUEUE TIME	XXXXXXX	MIN.

FILE #	MIN I/O	AVE I/O	MAX I/O
01	[illegible]	[illegible]	[illegible]
[illegible]			
[illegible]			
[illegible]			
[illegible]			
[illegible]			
[illegible]			
[illegible]			
[illegible]			
[illegible]			

A

W/R # MN NOT USED

[illegible]CATED = NNNN NUMBER OF TRANSACTIONS CALLED = [illegible]

MINIMUM		AVERAGE	MAXIMUM
XXXXXXX.	MIN.	XXXXXXX	XXXXXXX
XXXXXXX	MIN	XXXXXXX	XXXXXXX
XXXXXXX	MIN.	XXXXXXX	XXXXXXX
XXXXXXX	MIN.	XXXXXXX	[illegible]

I/O MAX I/O FILE [illegible] [illegible] I/O [illegible] I/O [illegible] I/O

[illegible] [illegible] [illegible] [illegible] [illegible]

B

SNAP

```
10      SNAP # NNNN   CPU   COT   AT   IOT      TIME        REC-CD
20                    XXX   XXX   XXX  XXX   XXXXX XXXX      XXXX

30      COT  W/R   CEN     ID     P-C  PCI    USI   IOT   TYPE
40           XXX XXX   XXXXXX    XX    XX   XXXXX  XXX   .XX   X
50           [illegible]      TIME-OFF-FIC        ADV-TIME
60           XXXXXX XXX       [illegible] XX      [illegible]
70           [illegible]  R-1   T-3  L-2  R-2    T-2  L-3 R-3
80           [illegible] XXXX XXXXXX XXXX XXXX XXXXX [illegible] XXXX X

90      AT   W/R   CEN     ID     P-C  PCI    USI   IOT   TYPE
40           XX  XXX  XXXXXX    XX    XX   XXXXX  XXX    XX   X
50           [illegible]      TIME-OFF-FIC        ADV-TIME
60           [illegible]      [illegible]         XXXXXX

100     IOT   TI   REC  TYPE   CPU  [illegible]  CYL   DIV   REN   OEN
110           XXX  XXX   XX     XX   XXXX   XXX   XXX   XXX   XXX
120           FILE  ID   DEV-LOC   BUFF    REC    [illegible]   TIME
130                 XXX  XXX   XXX  XXXX   XXX   [illegible]    XXXXX
140           DEV   ID   AVAIL  PLACE  T-I SIZE L-P CP [illegible]
150                 XXX . XXX . XXX   XXX  XXX XXX XX  XXXXX
160           CHAN  ID      RATE   INTER        [illegible]-TIME
170                 XXX  XXXXXXX  XXXXXX    XXXXX XXXXXX     XXX
```

NO

A

```
IOT     TIME          FCS-CD  ITEM  SUB  IOT
XXX  XXXXXXXXXX       XXXX    XXX   XXX

PRI   HSI    IOT   TYPE   LA-1  LC-1
 XX  XXXXX   XXX   .XX          XXXX
EFF-FLC       ADV-TIME         CPU-TIME        Q-SRT-TIME
XXXXXX                         XXXXXXXX        XXXXXXXX
D  R-2    T-2  B-3    F-3     T-3   E-4
X XXXX                                  XXXX

PRI   HSI    IOT   TYPE   LA-1  LC-1   LR-2
                     XX   XXXXX
EFF-FLC       ADV-TIME
                XXXXXX         XXXXXXXXXX      XXXXXXXX

CHAR  CYL   DEV   ARM   SIN   F-X        CHANNEL-END
XXXX  XXX   XXX   XXX   XXX   XXX
BUFF   REG
XXXX  XX
S  T-I  SIZE L-F  CP
K   XXX  XXX  XXX  XX  XXXXXX           XXXX
ENTER        ULE-TIME                  LOAD
XXXXX           XXXXXX       XXXX
```

NOT

B

QUEUE STATISTICS

QUEUE STATISTICS # NNNN

Q-NO.	TYPE	METHOD	MAXIMUM ENTRIES	CURRENT ENTR
NN	XX	XX	XXXX	XXXX

A

NNN

UM ENTRIES	CURRENT ENTRIES	AVERAGE ENTRIES	% UTILIZATION
XXXX	XXXX	XXXXX.XXX	XXX.XXX %

B

MEMORY STATISTICS

MEMORY STATISTICS # NNNN

MEMORY-NO.	NUMBER-OF-PACKS	NO-MEMORY-REFERENCES
XX	XXXXXXX	XXXXXXX

A

NNNN

NO-MEMORY-REFERENCES   CURRENT-PAGES-BUSY   AVERAGE-PAGES-BUSY   [illegible]

XXXXXXX   XXXX   [illegible]   [illegible]

B

INE STATISTICS

STI

FILE STATISTICS * NNNN

FILE-NO.	DEVICE-NO.	RELATIVE-LOCATION
XXX	XXX	NNN

TIME-RECEPTION   TOTAL-TIME-MIN   % TIME-SEIZED   [illegible]

XXXXX.XXX   X.X

NOT[illegible]DUC

b

...VICE STATISTICS

STM

DEVICE STATISTICS - HHHH

DEVICE-NO.	SEEK-TIME	% SEEK-TIME	PENALTY-TIM
XXX	XXXXX.XXX	XX.XXX %	[illegible]

A

SEEK-TIME	PENALTY-TIME	% PENALTY-TIME	USE TIME	% USE-TIME	[illegible]
XX.XX %	[illegible]	[illegible]	[illegible]	[illegible]	

CONTROL UNIT STATISTICS @ HHMM

CONTROL NO.	TIME-USED	% UTILIZATION
XX	XXXXX.XXX	XXXXX.XXX %

4

NNEL STATISTICS
ST6

CHANNEL STATISTICS # NNNN

CHANNEL-NO.	SELECTOR-USE-IN-MIN	% TIME USED
XX	XXXXX.XXX	XX.XXX %

A

[illegible]-IN-[illegible]	% TIME USED	% SELECTOR UTILIZATION	% MULTIPLEXOR [illegible]
[illegible].XXX	XX.XXX %	XX.XXX %	[illegible]

B

CPU + O/S
STATISTICS
S77

CPU #	O/S TI Q TIME	O/S IOTI Q TIME
XX	XXXX.XXX	XXXX.XXX
XX		
[illegible]		
[illegible]		
XX		
TOTALS		

A

TIME	O/S JOTE Q TIME	CYCLE TIME	[illegible] TIME	[illegible] TIME	[illegible]
XX	XXXX.XXX	XXXX.XXX	XXXX.XXX	XXX.XXX	XXX.XXX

B